RUPRECHT-KARLS-UNIVERSITÄT HEIDELBERG

Dimitri Probst

A Neural Implementation of Probabilistic Inference in Binary Probability Spaces

Masterarbeit

HD-KIP 14-31

KIRCHHOFF-INSTITUT FÜR PHYSIK

Faculty of Physics and Astronomy

University of Heidelberg

Master's thesis

in Physics

submitted by

Dimitri Probst

born in Zatobolsk, Kazakhstan

March 2014

A Neural Implementation of Probabilistic Inference in Binary Probability Spaces

This master's thesis has been carried out by Dimitri Probst at the

Kirchhoff Institute for Physics

Ruprecht-Karls-Universität Heidelberg

under the supervision of

Prof. Dr. Karlheinz Meier

A Neural Implementation of Probabilistic Inference in Binary Probability Spaces

Widely regarded as a hallmark of intelligence, be it artificial or biological, the ability to perform stochastic inference has been the subject of intense research in both the fields of machine learning and neuroscience. In this context, graphical models – such as Bayesian networks – provide a useful framework for representing probability distributions and performing inference in their respective probability spaces. Extending the theoretical approaches from *Pecevski et al.* [2011] and *Petrovici et al.* [2013], this thesis describes the "physical" implementation of arbitrary binary probability distributions, represented as Bayesian networks, in ensembles of leaky integrate-and-fire (LIF) neurons. Based on a sampling approach rather than belief propagation, the proposed implementation offers significant advantages in terms of sparseness, convergence and speed. In this framework, individual neurons represent the binary random variables, while conditional probabilities are embedded in the synaptic interactions, mediated by postsynaptic potentials (PSPs). Due to the difference between theoretically optimal PSP shapes and those achievable with LIF neurons, a novel interaction model is proposed, based on feedforward neural chains. This new approach is characterized in detail and validated through extensive software simulations. While creating a bridge to experimental neuroscience, the proposed approach inherently fosters a promising application for neuromorphic hardware, which can thereby provide the substrate for fast and power-efficient inference machines. As a necessary preliminary for such an application, several critical parameters of the BrainScaleS wafer-scale neuromorphic platform are characterized and discussed.

Eine neuronale Realisierung probabilistischer Inferenz in binären Wahrscheinlichkeitsräumen

Weitgehend aufgefasst als ein Kennzeichen von Intelligenz, sei sie künstlich oder biologisch, ist die Fähigkeit, stochastische Inferenz durchzuführen, Gegenstand umfangreicher Forschung sowohl im Bereich des maschinellen Lernens als auch der Neurowissenschaft. In diesem Zusammenhang bieten grafische Modelle – wie z.B. Bayes'sche Netze – einen hilfreichen Rahmen zur Darstellung von Wahrscheinlichkeitsverteilungen und Durchführung von Inferenz in den jeweiligen Wahrscheinlichkeitsräumen. Durch Erweitern der theoretischen Methoden von *Pecevski et al.* [2011] und *Petrovici et al.* [2013] beschreibt diese Arbeit die "physikalische"Realisierung beliebiger binärer Wahrscheinlichkeitsverteilungen, repräsentiert durch Bayes'sche Netze, in Ensembles von Leaky Integrate-and-Fire (LIF) Neuronen. Auf der Grundlage einer Samplingmethode an Stelle von Belief Propagation, bietet die vorgeschlagene Umsetzung wesentliche Vorteile hinsichtlich Effizienz, Konvergenz und Geschwindigkeit. In diesem Rahmen stellen einzelne Neuronen die binären Zufallsvariablen dar, während bedingte Wahrscheinlichkeiten in synaptischen Wechselwirkungen verankert sind und durch postsynaptische Potentiale (PSPs) vermittelt werden. Auf Grund der Abweichung der theoretisch optimalen von den mit LIF Neuronen erreichbaren PSP-Formen wird ein neuartiges Wechselwirkungsmodell vorgeschlagen, welches auf Feedforwardketten von Neuronen gründet. Dieser neuartige Ansatz wird eingehend charakterisiert und mittels umfassender Softwaresimulationen validiert. Indem es einen Übergang zur experimentellen Neurowissenschaft schafft, ermöglicht das vorgeschlagene Konzept eine vielversprechende Anwendung für neuromorphe Hardware, die dabei das Substrat für schnelle und leistungseffiziente Inferenzmaschinen bereitstellt. Als notwendige Vorbereitung für eine derartige Anwendung werden entscheidende Parameter der BrainScaleS wafer-skaligen neuromorphen Plattform charakterisiert und diskutiert.

Contents

1 Introduction

> We owe to the frailty of human mind
> one of the most delicate and ingenious
> of mathematical theories, namely the
> science of chance or probabilities.
>
> (*Pierre Simon Laplace* 1776)

The human mind is able to discover causal relationships in its surrounding world based only on a sparse amount of information and to use this knowledge to make predictive reasoning for future events. This is *the* key ability which makes it possible for our species to understand past occurrences and to plan and manipulate prospective actions. Most notably, the human mind can learn complex causalities with myriads of aspects and connections far beyond direct cause-effect relationships.
Inference in human mind seems to be Bayesian in nature, rather than purely frequentist. Decisions rarely depend only on observed events; usually, observations serve to update a prior model of the problem at hand. It therefore appears natural to describe human reasoning using Bayesian models. Indeed, Bayesian inference lies at the heart of most probabilistic models of cognition [*Griffiths et al.*, 2008]. In these approaches, the environmental situation, its causes and its possible effects are represented via random variables, and their underlying joint and conditional probability distributions. Two or more random variables can be causally related to each other. The resulting networks of random variables, so-called Bayesian networks, are an abstract diagrammatic representation of the causal structure of the contemplated problem.
The search for Bayesian inference at the level of neural network dynamics in the brain is novel compared the about 100 year-old history of neuroscience [*Knill and Pouget*, 2004; *Griffiths and Tenenbaum*, 2006; *Oaksford and Chater*, 2007; *Doya et al.*, 2011]. Only recently, empirical research started uncovering how the internal models of the mammalian brain progressively adapt to the statistics of natural stimuli at the level of single neurons [*Berkes et al.*, 2011]. Provided with support by these empirical studies, theoretical neuroscientists use computer simulations of spiking neural networks as a bottom-up approach for deciphering inference in the neocortex.
In the context of probabilistic inference, the question arises of how to represent statistical distributions at all. One possible way is to have a closed-form representation of all the relationships between the random variables and perform inference analytically. This idea builds the basis of powerful and often precise so-called belief propagation algorithms, which have also been investigated in the context of spiking neural networks [*Steimer et al.*, 2009; *Petkov*, 2012]. However, these algorithms often require complicated calculations and are not guaranteed to converge towards the correct result (loopy belief propagation,

see *Bishop* [2006]). In contrast to these analytical approaches, probabilistic distributions can be represented, up to an arbitrary degree of precision, by drawing samples from them. Sampling methods, such as Markov chain Monte Carlo (MCMC), have the advantage of being computationally efficient and, in particular, of being able to provide an increasingly improving approximation of the sought distribution at any moment during their application (anytime computing).

Early during the *BrainScaleS* project, a stochastic framework for probabilistic inference with spiking neural networks, the so-called *neural sampling* framework, has been developed, which combines MCMC sampling with the activity of neural networks [*Buesing et al.*, 2011; *Pecevski et al.*, 2011]. Each binary random variable is represented by a spiking neuron and the spiking frequency of the neuron is proportional to the probability of the corresponding variable to assume the state "1". *Buesing et al.* [2011] demonstrate that the sequences of states assumed by a network of stochastic neurons can be used to sample from distributions with second-order dependencies over binary variables (Boltzmann distributions). *Pecevski et al.* [2011] extend this approach to arbitrary distributions, which they represent as Bayesian graphical models.

The original neural sampling algorithm uses an abstract neuron model with a sigmoidal activation function, non-resetting membrane potential after the emission of a spike, an additional so-called refractory variable and rectangular postsynaptic potentials (PSPs), none of which applies to biological or standard neuron models. *Petrovici et al.* [2013], however, demonstrate that the abstract neuron model from *Buesing et al.* [2011] can be mapped to a deterministic Leaky Integrate-and-Fire (LIF) neuron embedded in a noisy environment.

The aim of this master's thesis is to extend the framework from *Petrovici et al.* [2013] by transferring the implementation of Bayesian networks, proposed by *Pecevski et al.* [2011], to networks of LIF neurons. In particular, the more demanding requirements imposed by the additional complexity of Bayesian networks as compared to Boltzmann machines require additional structural elements, which are discussed in detail throughout this work. Beyond investigating the compatibility of the theory proposed by *Pecevski et al.* [2011] and networks of LIF neurons, extensive validation runs in software are performed against the benchmark provided by the abstract neuron model. On the long run, however, the modeling of large Bayesian networks with the help of software simulations will become unsuitable because the duration of the simulation increases exponentially with the size of the neural network.

The *BrainScaleS* neuromorphic hardware system [*Schemmel et al.*, 2010], which is being developed in cooperation of the *Electronic Vision(s)* group at the University of Heidelberg and the group for Parallel VLSI Systems and Neural Circuits at the TU Dresden, offers a solution for these time-consuming experiments. The physical modeling of neural networks on a neuromorphic hardware substrate comes with multiple crucial advantages compared to traditional numerical simulations. First, neuromorphic systems can be characterized by a massive parallelism which is only present in the most efficient computing system in the universe, the human brain. Together with the inherent time scales of the silicon substrate, this aspect allows to speedup experiments up to 4 orders of magnitude faster than biological real time, and up to 6 orders of magnitude faster than nowadays' computer

simulations. Second, the emulation of the neuron's differential equations only involves a few transistors leading to a power consumption reduction by several orders of magnitude compared to numerical simulations [*Schemmel et al.*, 2010]. The inherent reduction of the power consumption fosters excellent scalability, with the potential to emulate large cortical areas, or even an entire brain.
On that account, the final part of this manuscript is dedicated to extensive tests of the compatibility of the sampling framework from *Petrovici et al.* [2013] with the neural and synaptic parameters of the *BrainScaleS* neuromorphic hardware system.

Outline

This manuscript is structured hierarchically, with each new topic building on the previously discussed ones. For a detailed understanding of the presented arguments, reading the chapters in their given order is recommended.
Chapter 2 outlines the mathematical and experimental prerequisites in detail. In particular, the theory of sampling with spiking neurons, on the one hand, and the hardware and software framework which are used to implement the theory, on the other hand, are in the main focus. Chapter 3 provides simulation results of sampling from Boltzmann distributions with LIF neurons. This chapter functions as a preparatory study for the remaining experiments described in this thesis. Chapters 4 and 5 represent the backbone of this manuscript. There, two different implementations of Bayesian networks proposed by *Pecevski et al.* [2011] are discussed in detail and translated to networks of LIF neurons. Additional network substructures required by the LIF implementation are designed and characterized, and the sampling performance of these networks is investigated against the benchmark provided by the theoretically ideal abstract model. Chapter 6 illustrates experimental results aiming to characterize the neuromorphic hardware substrate described in Chapter 2 and, in particular, to test its compatibility with the requirements of the sampling framework. In addition, software simulations are used to spot the parameter ranges which entail the potential to improve the quality of future sampling experiments on the neuromorphic hardware substrate. Chapter 7 gives a summary of the results achieved in this thesis. Chapter 8 finally concludes the manuscript by listing suggestions for prospective experiments which will enhance the sampling results in both software and hardware implementations.

2 Materials and Methods

This chapter provides an overview, on the one hand, of the theoretical models utilized throughout the thesis and, on the other hand, of the tools which were used to implement the theoretical models.
Broadly speaking, this thesis focuses on the implementation of an abstract theoretical model in a neuromorphic substrate. The largest part of this chapter is therefore dedicated to a step-by-step introduction of the theoretical background (see Section 2.1). Here, we start by summarizing the most essential concepts of probability theory. Afterwards, two powerful classes of physical implementations of probability distributions are presented: Boltzmann machines and Bayesian networks. Incidentally[1], these are widely used for modeling paradigms like probabilistic inference in human reasoning (see e.g. *Alais and Blake* [2005] or *Knill and Kersten* [1991]).
An analytical evaluation of probabilities is not always computationally feasible. However, under certain conditions, sampling algorithms offer an efficient alternative for representing distributions with, in principle, an arbitrary degree of precision.
Of particular interest to us is the theory of *neural sampling*, which creates a link between the dynamics of spiking networks and the widely used Gibbs sampling algorithm. Here, we briefly recap the neural sampling approach from *Buesing et al.* [2011] and discuss concrete implementations of Boltzmann machines and Bayesian networks with stochastic spiking neural networks.
Neural sampling ultimately serves as a basis for *LIF sampling*, which denotes sampling from probability distributions with deterministic LIF neurons in a spiking noisy environment. Here, we first describe the membrane dynamics of LIF neurons in detail and observe that they can be well characterized by an *Ornstein-Uhlenbeck* process under certain stimulus conditions. This equivalence is the prerequisite for proving that LIF neurons can be used to sample from well-defined probability distributions.
Section 2.2 surveys the BrainScaleS wafer-scale hardware system, which is the neuromorphic hardware substrate on which the feasibility of the LIF sampling implementation was tested in the course of this thesis. Here, we begin with the description of the analog and digital circuitry of the HICANN chip, which is the centerpiece of the hardware system. Thereafter, we continue with the demonstrator setup, which entails the identical communication infrastructure to the HICANN chip as the actual wafer-scale system and which was used to run experiments on the HICANN chip. Finally, the prospective Hybrid Multiscale Facility (HMF) will be briefly introduced.

[1] While Boltzmann machines have been inspired by Hopfield networks [*Hopfield*, 1982], which in turn have been designed as abstract replicas of biological content-addressable memory [*Sejnowski*, 1986], Bayesian networks were first developed within the machine learning community [*Pearl*, 1985] and only later adopted to explain how animals can perform inference (see e.g. *Oaksford and Chater* [2007]).

Section 2.3 concludes this chapter by presenting the software tools which were applied to setup neural networks and conduct software and hardware experiments throughout this thesis. This section is divided into two subparts. The first part describes the simulator-independent modeling tool PyNN which will be used to run the experiments with deterministic LIF neurons in Chapters 3, 4 and 5. The second part introduces the the native interpreter of the BrainScaleS wafer-scale hardware system, the so-called Hardware Abstraction Layer (HAL). The Python-based PyHAL API, which wraps the Hardware Abstraction Layer (HAL) interface, is used to setup the hardware experiments in Chapter 6.

2.1 Theoretical Prerequisites

This section will introduce the theoretical methods underlying this thesis. Subsection 2.1.1 offers a compact overview of the basics of probability theory. Afterwards, in 2.1.2 and 2.1.3, two important representations of probability distributions are presented, namely Boltzmann machines and Bayesian networks. Subsection 2.1.4 introduces MCMC sampling in general and *Gibbs sampling* as a special case, which allows sampling from probability distributions without knowing their partition function. The MCMC framework is instrumental for explaining *neural sampling*, which is presented in Subsection 2.1.5. Thereafter, Subsection 2.1.6 discusses the dynamics of the utilized deterministic neuron and synapse models. Subsection 2.1.7 finally transfers neural sampling to leaky integrate-and-fire neurons, referred to as *LIF sampling*.

2.1.1 Basics of Probability Theory and Inference

The following summary of basic principles of probability theory is based on *Bishop* [2006] and *Griffiths et al.* [2008].
A *random variable* (RV) X describes the outcome of a random event. For each RV X and member c of a set of real numbers, one can calculate the *probability* $p\,(X = c)$ that X takes the value c. The collection of all these probabilities results in the *distribution of* X. A RV can be *discrete* or *continuous*. Discrete RVs can assume a nonnegative probability for a countable set of values. A discrete probability distribution is described by its *probability mass function* $p\,(X = c)$ for which the following rules have to be fulfilled:

$$0 \leq p\,(X = c) \leq 1\ , \tag{2.1}$$

and

$$\sum_{c} p\,(X = c) = 1\ . \tag{2.2}$$

If X is a continuous RV, it has an associated *probability density function* $f(x)$, which satisfies:

$$p\,(a < X \leq b) = \int_{a}^{b} f(x)\,\mathrm{d}x\ , \tag{2.3}$$

and

$$\int_{-\infty}^{\infty} f(x)\,\mathrm{d}x = 1\ . \tag{2.4}$$

Binary RVs are in the main focus of this thesis. A binary RV X can assume the values $c \in \{0, 1\}$.
Multiple RVs $X_1, X_2, X_3, ..., X_K$ can be combined to a random vector $\mathbf{X} = (X_1, X_2, X_3, ..., X_K)$ with a *multivariate distribution* over all random variables $p(\mathbf{x} = (x_1, x_2, x_3, ..., x_K))$ with possible assignments $x_1, x_2, x_3, ..., x_K$ of the variables. $p(\mathbf{x})$ is called the *joint probability distribution* of the RVs $X_1, X_2, X_3, ..., X_K$.
The probability distribution associated with only one of the variables, e.g. X_k, is called a *marginal probability distribution*. It is calculated by summing the joint distribution over all possible assignments of all the other variables, which is known as *marginalization*:

$$p(x_k) = \sum_{x_1} ... \sum_{x_{k-1}} \sum_{x_{k+1}} ... \sum_{x_K} p(\mathbf{x})\ . \tag{2.5}$$

If some of the RVs $X_{k+1}, ..., X_K$ are known to assume the particular values $x_{k+1}, ..., x_K$, then the joint probability distribution $p(\mathbf{x})$ can be factorized into

$$p(x_1, ..., x_K) = p(x_1, ..., x_k | x_{k+1}, ..., x_K)\, p(x_{k+1}, ..., x_K)\ , \tag{2.6}$$

where $p(x_1, ..., x_k | x_{k+1}, ..., x_K)$ is the *conditional probability* of $x_1, ..., x_k$ given $x_{k+1}, ..., x_K$.
Equation 2.6 ultimately results in what is known as *Bayes' rule*:

$$p(x_1, ..., x_k | x_{k+1}, ..., x_K) = \frac{p(x_{k+1}, ..., x_K | x_1, ..., x_k)\, p(x_1, ..., x_k)}{p(x_{k+1}, ..., x_K)}\ . \tag{2.7}$$

The four probability distributions in Equation 2.7 are referred to as

- the *prior* probability distribution $p(x_1, ..., x_k)$,
- the *posterior* probability distribution $p(x_1, ..., x_k | x_{k+1}, ..., x_K)$,
- the *likelihood* function $p(x_{k+1}, ..., x_K | x_1, ..., x_k)$,
- the *evidence* $p(x_{k+1}, ..., x_K)$.

The derivation of the posterior probability from the prior probability and the likelihood function of the assumed model is known as *probabilistic inference*.
Schematic representations of probability distributions, so-called *probabilistic graphical models*, are helpful if complex inferential computations have to be performed. In probabilistic graphical models, RVs are represented by *nodes* and probabilistic relationships between RVs are expressed by *edges* [*Bishop*, 2006]. Figure 2.1 illustrates examples of the two general classes of probabilistic graphical models, namely *Markov random fields* (or *undirected graphical models*) and *Bayesian networks* (or *directed graphical models*). Markov random fields are useful for expressing soft constraints between RVs, while Bayesian networks are suited to expressing causal relationships between RVs [*Bishop*,

A: Markov random field **B:** Bayesian network

Figure 2.1: Examples of probabilistic graphical models of a probability distribution of 3 RVs. **(A)** Markov random field which describes the probability distribution $p(x_1, x_2, x_3) = \frac{1}{Z}\phi_1(x_1, x_3)\phi_2(x_2, x_3)$ with some nonnegative specific potential functions ϕ_1 and ϕ_2 and the partition function Z, which ensures normalization. **(B)** Bayesian network which represents the probability distribution $p(x_1, x_2, x_3) = p(x_1)\, p(x_2)\, p(x_3|x_1, x_2)$.

2006]. Section 2.1.2 will describe one special type of Markov random fields, so-called *Boltzmann machines*. Afterwards, Section 2.1.3 will introduce Bayesian networks.
A typical evaluation for probabilistic inference in a graphical model which represents a multivariate distribution $p(\mathbf{x})$ is the computation of a conditional probability $p(x_1, ..., x_k|x_{k+1}, ..., x_K)$ or marginals thereof. The values $x_{k+1}, ..., x_K$, which are fixed, could e.g. represent some sensory input or a specific goal for a decided motoric action, and $x_1, ..., x_l$ have to be inferred from them.
Computing a marginal $p(x_n)$ with a brute force summation over all $K-1$ remaining RVs via Equation 2.5 would require the evaluation and storage of 2^{K-1} distinct values in the binary case, so that quickly rendering marginalizations in large ensembles becomes computationally infeasible. Broadly speaking, one can distinguish between two conceptually different approaches to this problem:

- *Exact* methods exploit the structure of the underlying graphical model. *Message-passing algorithms* are a widely used example of exact inference. Message-passing algorithms are based on clever reorderings of sums and products of the joint probability distribution during marginalization. Intermediate sums that arise during the calculation can be viewed as *messages* attached to the edges in the graphical model. In this context, inference can be viewed in terms of a local computation and routing of messages [*Arbib*, 2002]. Message-passing algorithms provide an exact solution to the inference problem if the graphical model does not contain any loops. In many other cases, they yield an approximate solution. *Steimer et al.* [2009] and *Petkov* [2012] extensively delve into exact computations of marginal and conditional probabilities via abstract stochastic neurons or via deterministic spiking neurons, respectively.

- *Approximate* methods exploit the *law of large numbers*, which describes that the relative frequencies of the results of a random experiment converge to the expected probability distribution if performing the experiment a large number of times.

Monte Carlo algorithms are characteristic for approximate methods. Monte Carlo algorithms are based on the fact that while it may not be possible to compute expectations under $p(\mathbf{x})$, it may be feasible to obtain samples from $p(\mathbf{x})$, or from a closely related distribution [*Arbib*, 2002]. Marginals and other expectations can then be approximated using sample-based averages. *Markov chain Monte Carlo* (MCMC) and *Gibbs sampling* as a special case of MCMC are examples for Monte Carlo algorithms (see Section 2.1.4).

2.1.2 Boltzmann Machines

Boltzmann Machines (BMs) are a special type of Markov random fields [*Ackley et al.*, 1985]. In its original meaning, a BM describes a physical instantiation of the Boltzmann distribution which is a network of symmetrically connected binary units that are randomly either "on" (1) or "off" (0) [*Hinton*, 2007]. For a binary random vector $\mathbf{Z} = (Z_1, ..., Z_K)$, the general shape of the Boltzmann distribution is

$$p(\mathbf{z}) = \frac{1}{Z} \exp\left(\sum_{i,j} \frac{1}{2} W_{ij} z_i z_j + \sum_i b_i z_i\right) \tag{2.8}$$

with arbitrary real-valued parameters b_i and W_{ij}, which satisfy the conditions $W_{ij} = W_{ji}$ and $W_{ii} = 0$. The parameter b_i is called the *bias* of Z_i while W_{ij} is the *weight* between Z_i and Z_j. The constant Z ensures normalization and is called the *partition function* of $p(\mathbf{z})$:

$$Z = \sum_{\mathbf{z}} \exp\left(\sum_{i,j} \frac{1}{2} W_{ij} z_i z_j + \sum_i b_i z_i\right) . \tag{2.9}$$

Each state is associated with an *energy function*

$$E(\mathbf{z}) = -\sum_{i,j} \frac{1}{2} W_{ij} z_i z_j - \sum_i b_i z_i . \tag{2.10}$$

From Equation 2.8 and 2.10 it follows immediately that states with lower energies are more likely.
By definition (see Equation 2.8), a Boltzmann distribution maximally allows pairwise, or *second-order*, interactions between RVs. Two RVs which are not connected to each other do not interact. Second-order interactions between RVs, however, are not adequate to model numerous computational tasks in the brain (see e.g. Figure 2.2).
One example are so-called *explaining away* effects. For instance, the graphical model in Figure 2.1B can be interpreted as a probability distribution in which X_1 and X_2 model two competing causes, or hypotheses, for the random occurrence of the event described by X_3. A change in the probability of one of the hypotheses would affect the probability of the other hypothesis, even though both of them are not causally related. These higher-order interactions between RVs can be described with Bayesian networks, which will be presented in the following section.

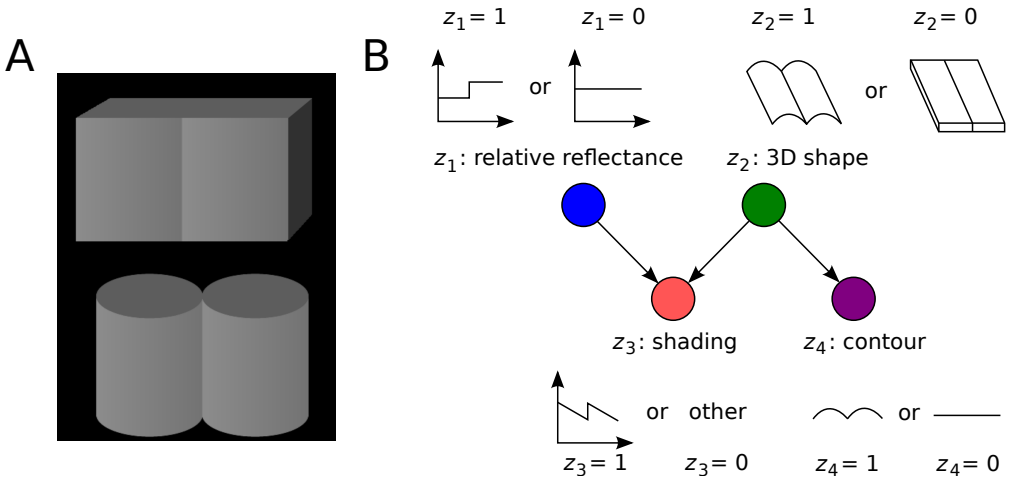


Figure 2.2: Demonstration of the *explaining away* effect in the visual perception experiment from *Knill and Kersten* [1991]. Panel **A** shows the phenomenon. Two visual stimuli originate from the reflectance of two geometrical objects which are both composed of two identical 3D shapes. Both stimuli feature the same shading profile in the horizontal direction. The perception of the reflectance of each stimulus is influenced by the perceived 3D shape: In the case of a flat contour, the right subobject appears brighter than the left one. This reflectance step is hardly observable for a cylindrical contour. A cylindrical 3D shape thus *explains away* the reflectance step. Panel **B** demonstrates the mathematical description of this optical illusion. The corresponding Bayesian network model consists of four RVs: z_1 (reflectance step versus uniform reflectance), z_2 (cylindrical versus flat 3D shape), z_3 (sawtooth-shaped shading profile versus some other profile) and z_4 (cylindrical versus flat contour). The inference of the probability distribution $p\left(z_1, z_2 | z_3 = 1, z_4 = 0\right)$ models the perception of the upper stimulus of Panel A, while the lower stimulus is represented by the inference of the probability distribution $p\left(z_1, z_2 | z_3 = 1, z_4 = 1\right)$. The figure is taken from *Pecevski et al.* [2011].

2.1.3 Bayesian Networks

A Bayesian Network (BN) is a *directed acyclic* graphical model whose nodes represent the RVs $Z_1, ..., Z_K$ [*Bishop*, 2006; *Koller and Friedman*, 2009]. The joint distribution defined by a Bayesian graph is given by the product of a conditional distribution for each node conditioned on the parent variables of that node. For a graph with K nodes, the joint probability distribution is given by

$$p(\mathbf{z}) = \prod_{k=1}^{K} p(z_k | \mathrm{pa}_k) \ , \tag{2.11}$$

where pa_k expresses the set of parents of z_k. The factor $p(z_k|\mathrm{pa}_k)$ is called an n^{th}-*order factor* if it depends on n RVs or rather $|\mathrm{pa}_k| = n - 1$.

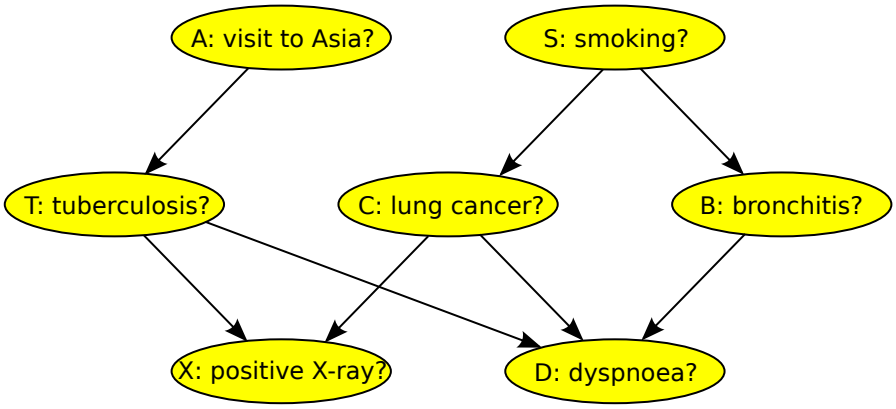
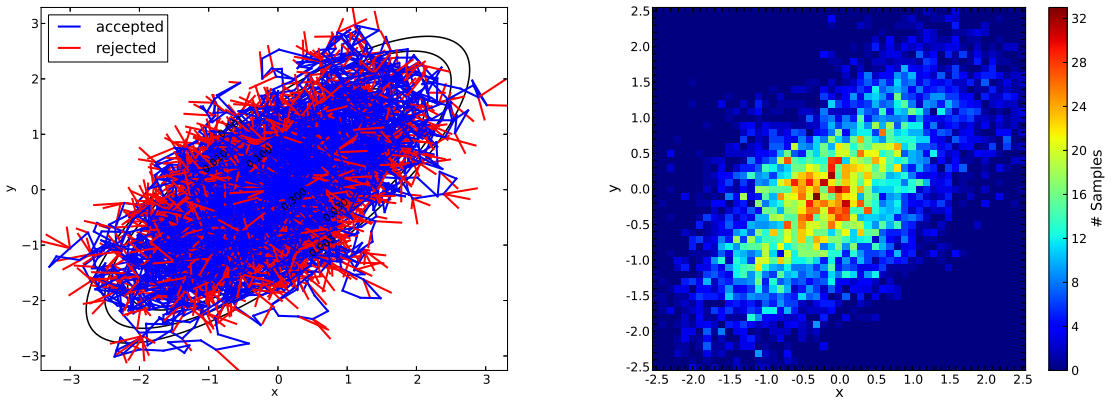


Figure 2.3: A simplified version of the ASIA Bayesian network from *Lauritzen and Spiegelhalter* [1988]. The graph describes the situation of a doctor who has to infer which of three diseases tuberculosis (T), lung cancer (C) and bronchitis (B) his patient has based on some contextual background information and on medical indicators. The 7 RVs are arranged in a three-layer causal structure. The 2 RVs A and S provide information whether the patient has recently visited Asia (A) or is a smoker (S). The observable symptoms are the result of an X-ray test (X) and dyspnoea (D). The graph contains two explaining away effects and undirected cycles.

Figure 2.2 depicts an example of an optical illusion in Panel A and its Bayesian network model in Panel B. The network models a visual perception experiment which was first demonstrated by *Knill and Kersten* [1991]. Panel A shows the visual stimuli of two geometrical objects which are both composed of two identical 3D shapes. Both stimuli feature the same shading profile in the horizontal direction, but differ in their contours. The perception of the reflectance of each stimulus is influenced by the perceived 3D shape: In the case of a flat contour, the right subobject appears brighter than the left one. This reflectance step is hardly observable in the case of a cylindrical contour. A cylindrical 3D shape thus *explains away* the reflectance step. Conversely, a reflectance step *explains away* the presence of a cylindrical 3D shape.
Panel B shows the corresponding Bayesian network. The model consists of four RVs: z_1 (reflectance step versus uniform reflectance), z_2 (cylindrical versus flat 3D shape), z_3 (sawtooth-shaped shading profile versus some other profile) and z_4 (cylindrical versus flat contour). The joint probability of these RVs is given by the network structure:

$$p\left(z_1, z_2, z_3, z_4\right) = p\left(z_1\right)\, p\left(z_2\right)\, p\left(z_3|z_1, z_2\right)\, p\left(z_4|z_2\right)\ . \tag{2.12}$$

The inference of the probability distribution $p\left(z_1, z_2|z_3 = 1, z_4 = 0\right)$ models the perception of the upper stimulus of Panel A, while the lower stimulus is represented by the inference of the probability distribution $p\left(z_1, z_2|z_3 = 1, z_4 = 1\right)$.
Figure 2.3 illustrates another example of a Bayesian network which is a modified version of the *ASIA network* initially introduced in *Lauritzen and Spiegelhalter* [1988]. The network



A: Sampling transitions **B:** Resulting sampling distribution

Figure 2.4: Demonstration of the Metropolis-Hastings algorithm **(A)** in two dimensions with a uniform proposal distribution q and a Gaussian target distribution p [*Bishop*, 2006]. Iso-probability lines of the target distribution are shown in *black*. Samples are drawn from the proposal distribution q and accepted according to Equation 2.14. Accepted transitions are shown in *blue*, while rejected ones are *red*. In this case already 5000 drawn samples can well approximate the target distribution **(B)**.

contains seven RVs: two context aspects (A: visit to Asia, S: smoking), three diseases (T: tuberculosis, C: lung cancer, B: bronchitis) and two indicators (X: positive X-ray test, D: dyspnoea). The joint probability of the network is given by

$$p(A,S,T,C,B,X,D) = p(A)\; p(S)\; p(T|A)\; p(C|S)\; p(B|S)\; p(X|T,C)\; p(D|T,C,B)\;. \tag{2.13}$$

The graph shows a more complex structure than the previous example, i.e. two *explaining away* effects and undirected cycles, e.g. between the RVs S, C, D and B. A typical example for evidence in this network is the knowledge that the person has recently visited Asia ($A = 1$) and exhibits the symptom of breathlessness ($D = 1$). The inference task is to calculate the likelihoods of the three diseases tuberculosis, lung cancer and bronchitis and how a positive X-ray test ($X = 1$) would affect these likelihoods [*Lauritzen and Spiegelhalter*, 1988].

2.1.4 Markov Chain Monte Carlo Sampling

Markov chain Monte Carlo (MCMC) sampling can be used to construct samples $\mathbf{z}$ from probability distributions $p(\mathbf{z})$ even if the normalizing constant is unknown. MCMC methods produce a new sample via a *local* search around a sample from the distribution rather than a global search over the whole state space of the RVs [*Bishop*, 2006].

Metropolis-Hastings algorithm: Each MCMC method uses an arbitrary proposal distribution $q\left(\mathbf{z}|\mathbf{z}^{(\tau)}\right)$ to generate a sample based only on the current sample $\mathbf{z}^{(\tau)}$, where τ is the iteration step. At each iteration, a candidate sample $\mathbf{z}^*$ is drawn from the proposal distribution. The *Metropolis-Hastings* algorithm then offers a general criterion, according to which this candidate sample $\mathbf{z}^*$ is accepted with probability

$$A\left(\mathbf{z}^*, \mathbf{z}^{(\tau)}\right) = \min\left(1, \frac{\tilde{p}(\mathbf{z}^*)q(\mathbf{z}^{(\tau)}|\mathbf{z}^*)}{\tilde{p}(\mathbf{z}^{(\tau)})q(\mathbf{z}^*|\mathbf{z}^{(\tau)})}\right) , \tag{2.14}$$

with $p\left(\mathbf{z}\right) = \tilde{p}\left(\mathbf{z}\right)/Z_p$ and the normalizing constant Z_p [*Bishop*, 2006]. If the candidate sample is accepted, then $\mathbf{z}^{(\tau+1)} = \mathbf{z}^*$, otherwise it is rejected and $\mathbf{z}^{(\tau+1)} = \mathbf{z}^{(\tau)}$. The next sample is drawn from $q\left(\mathbf{z}|\mathbf{z}^{(\tau+1)}\right)$. Figure 2.4 shows an example of the Metropolis-Hastings algorithm in two dimensions.

Gibbs sampling: The *Gibbs sampling* algorithm is a special case of the Metropolis-Hastings algorithm [*Geman and Geman*, 1984]. At each iteration of the algorithm, the value z_k of one of the RVs from the random vector $\mathbf{Z}$ is replaced by a value which is drawn from the proposal distribution $p\left(z_k|\mathbf{z}_{\setminus k}\right)$. The vector $\mathbf{z}_{\setminus k}$ contains all assignments $z_1, ..., z_K$ but with z_k omitted. This update scheme is repeated for all RVs of the vector $\mathbf{Z}$, either in some defined order or at random [*Bishop*, 2006]. With ${\mathbf{z}^*}_{\setminus k} = \mathbf{z}_{\setminus k}$ and $p\left(\mathbf{z}\right) = p\left(z_k|\mathbf{z}_{\setminus k}\right) p\left(\mathbf{z}_{\setminus k}\right)$ in Equation 2.14, the acceptance probability of the Gibbs sampling algorithm is always 1 by design.
For the concrete example of a probability distribution $p\left(z_1, z_2, z_3\right)$ of three RVs z_1, z_2, z_3 with initial samples $z_1^{(0)}, z_2^{(0)}, z_3^{(0)}$, the first sample is drawn from $p(z_1^{(1)}|z_2^{(0)}, z_3^{(0)})$, the following one from $p(z_2^{(1)}|z_1^{(1)}, z_3^{(0)})$ and so on.

Kullback-Leibler divergence: The approximation quality of the sampled probability distribution can be evaluated by a measure of the difference of the approximated probability distribution $q(\mathbf{z})$ and the target probability distribution $p(\mathbf{z})$, the so-called *Kullback-Leibler* (KL) divergence $D_{\mathrm{KL}}(q||p)$. It is defined as

$$D_{\mathrm{KL}}(q||p) = \sum_{\mathbf{z}} q(\mathbf{z}) \log\left(\frac{q(\mathbf{z})}{p(\mathbf{z})}\right) . \tag{2.15}$$

The KL divergence is a distance measure $D_{\mathrm{KL}}(q||p) \geq 0$, with equality if and only if $p = q$. But, unlike a distance, it is not symmetric with respect to interchange of p and q [*Dayan and Abbott*, 2001].

Markov chain: MCMC algorithms create a set of consecutive samples $\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, ...$ which forms a *Markov chain* of order 1. A *Markov chain of order m* is defined by a sequence of states $\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, ..., \mathbf{z}^{(K)}$ of a random vector $\mathbf{Z}$ such that the following conditional independence property holds for $k \in \{1, ..., K-1\}$:

$$p\left(\mathbf{z}^{(k+1)}|\mathbf{z}^{(k)}, \mathbf{z}^{(k-1)}, ..., \mathbf{z}^{(1)}\right) = p\left(\mathbf{z}^{(k+1)}|\mathbf{z}^{(k)}, \mathbf{z}^{(k-1)}, ..., \mathbf{z}^{(k-m+1)}\right) , \tag{2.16}$$

with $k \geq m$. For the particular example of $m = 1$, a Markov chain can be characterized by the probability distribution for the initial state $p(\mathbf{z}^{(0)})$ and a *transition operator* $T\left(\mathbf{z}^{(k)}, \mathbf{z}^{(k+1)}\right) \equiv p\left(\mathbf{z}^{(k+1)}|\mathbf{z}^{(k)}\right)$ [*Bishop*, 2006]. The chain starts in some initial state $\mathbf{z}^{(0)}$ and moves through a trajectory of states $\mathbf{z}^{(\tau)}$ drawn from the conditional probability distribution $T\left(\mathbf{z}^{(\tau)}, \mathbf{z}^{(\tau+1)}\right)$.
A Markov chain can have several important properties. It is called *irreducible* if any state $\mathbf{z}^{(k+1)}$ can be reached from any other state $\mathbf{z}^{(k)}$ in finitely many steps with a probability larger than zero. A Markov chain is *aperiodic* if its state transitions cannot be trapped in deterministic cycles. Irreducibility and aperiodicity are sufficient conditions for ensuring that the required distribution $p(\mathbf{z})$ is *invariant* or *stationary*, i.e. that the probability distribution $p\left(\mathbf{z}^{(\tau)}|\mathbf{z}^{(0)}\right)$ converges for $\tau \to \infty$ to the distribution $p(\mathbf{z})$ that does not depend on the initial state $\mathbf{z}^{(0)}$ [*Grimmett and Stirzaker*, 2001]. A Markov chain of order 1 is said to be *reversible* if its transition operator T satisfies the *detailed balance* condition:

$$T\left(\mathbf{z}^{(k+1)}, \mathbf{z}^{(k)}\right) p\left(\mathbf{z}^{(k+1)}\right) = T\left(\mathbf{z}^{(k)}, \mathbf{z}^{(k+1)}\right) p\left(\mathbf{z}^{(k)}\right) \ . \tag{2.17}$$

Reversibility is a sufficient but not necessary condition for the invariance of the probability distribution $p(\mathbf{z})$ [*Bishop*, 2006]. It is important to note that indeed many physical ensembles, e.g. neural networks, have irreversible dynamics (see Section 2.1.5).

2.1.5 Neural Sampling

The term *neural sampling* in general refers to sampling from probability distributions with networks of spiking neurons. This section presents the theory of neural sampling from *Buesing et al.* [2011] which links MCMC sampling to the dynamics of spiking neurons. In the context of *Buesing et al.* [2011], neural sampling is performed with a particular abstract inherently stochastic neuron model.
In contrast to MCMC, spiking neurons do not incorporate reversible dynamics due to refractory mechanisms. For example, a neuron which is not refractory can always be brought into the refractory state with enough stimulation, which is not possible for the opposite case.
However, *Buesing et al.* [2011] prove that the Markov chain built into the dynamics of a network of stochastic neurons can be used to sample from probability distributions. This offers the theoretical framework for the physical instantiation of Boltzmann machines and Bayesian networks with spiking neurons.

Neural Sampling in Discrete and Continuous Time: In the abstract model used by *Buesing et al.* [2011], a neuron elicits spikes stochastically depending on the current membrane potential. The assignment z_k of a RV Z_k from a binary random vector $\mathbf{Z}$ is encoded in the spiking activity of a neuron ν_k. If the neuron ν_k has elicited a spike within the recent τ time steps (*refractory period*) it encodes the state $z_k = 1$, otherwise $z_k = 0$. The time since the last spike is measured by an additional non-binary *refractory variable* $\zeta_k \in \{0, 1, ..., \tau\}$ for each individual neuron ν_k.

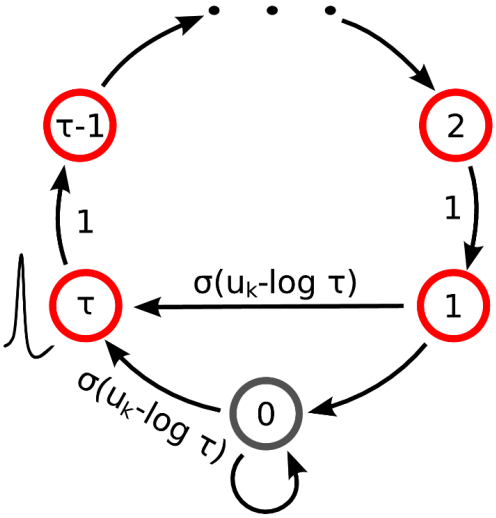


Figure 2.5: Illustration of the local transition operator T^k for the internal state variable ζ_k of a neuron ν_k from *Buesing et al.* [2011]. The transition probability to the state ζ_k depends only on the previous state ζ'_k. The neuron is allowed to elicit a spike for $\zeta_k \leq 1$.

In order for a network to sample correctly from a target distribution $p\left(z_1, ..., z_K\right)$ over the binary random vector $\mathbf{Z}$, each of its constituent neurons ν_k must "know" their respective conditional probability $p\left(z_k|\mathbf{z}_{\setminus k}\right)$. The so-called *neural computability condition* (NCC) provides a sufficient condition for correct sampling, wherein a neuron's "knowledge" about the state of the rest of the network is encoded in its membrane potential:

$$u_k(t) = \log \frac{p(z_k(t) = 1|\mathbf{z}_{\setminus k}(t))}{p(z_k(t) = 0|\mathbf{z}_{\setminus k}(t))} \,, \tag{2.18}$$

where $\mathbf{z}_{\setminus k}(t)$ are the current values $z_i(t)$ of all other variables z_i with $i \neq k$. It is required that each single neuron in the network fulfills the NCC. Equation 2.18 can also be formulated in terms of an *activation function*

$$p(z_k(t) = 1|\mathbf{z}_{\setminus k}(t)) = \sigma\left(u_k\left(t\right)\right) := \frac{1}{1 + \exp\left(-u_k(t)\right)} \,, \tag{2.19}$$

exploiting the condition $p(z_k = 1) = 1 - p(z_k = 0)$.
The neural dynamics can be expressed via a *local transition operator*

$$T^k(\zeta_k|\zeta'_k, \mathbf{z}_{\setminus k}) = \begin{cases} \sigma(u_k - \log\tau), & \text{if } \zeta_k = \tau \text{ and } \zeta'_k \leq 1 \\ 1 - \sigma(u_k - \log\tau), & \text{if } \zeta_k = 0 \text{ and } \zeta'_k \leq 1 \\ 1, & \text{if } \zeta_k = \zeta'_k - 1 \text{ and } \zeta'_k > 1 \\ 0, & \text{otherwise} \end{cases} \,, \tag{2.20}$$

and

$$z_k = \begin{cases} 1, & \text{if } \zeta_k > 0 \\ 0, & \text{if } \zeta_k = 0 \end{cases} \,. \tag{2.21}$$

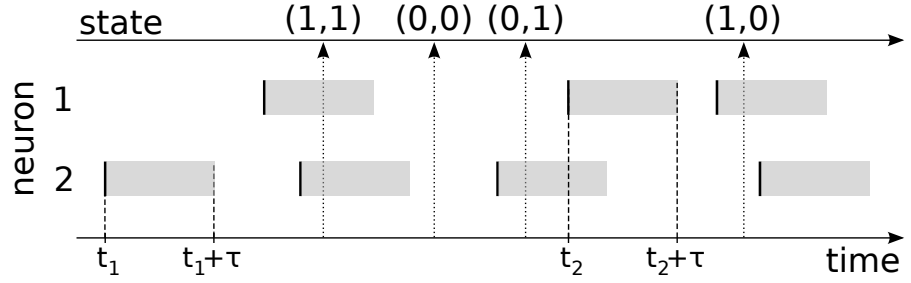


Figure 2.6: The spike pattern of two neurons which are sampling from a probability distribution $p(z_1, z_2)$. The neurons have the refractory period τ (*gray box*) during which their associated RVs keep the state $z_k = 1$. The overall network state at time t is (z_1, z_2).

The transition probability to the state ζ_k only depends on the previous state ζ'_k. The resulting sequence of states $\zeta_k(t=0), \zeta_k(t=1), \zeta_k(t=2), ...$ is a Markov chain. Equation 2.20 implements what is known as *Glauber dynamics*. Figure 2.5 illustrates the dynamics of Equation 2.20.
Buesing et al. [2011] prove that after some (ideally infinite) *burn-in* time, the dynamics of the network given by the transition operator T^k produce samples from the extended distribution $p(\zeta, \mathbf{z})$. The distribution $p(\mathbf{z})$ is provided by marginalizing over ζ: $p(\mathbf{z}) = \sum_\zeta p(\zeta, \mathbf{z})$. Thus, given that each single neuron in the network fulfills the NCC, the network will sample from the target distribution $p(\mathbf{z})$.
The spiking neural network can also be used to sample from the posterior distribution $p(z_1, ..., z_k | z_{k+1}, ..., z_K)$ with the observed subset of variables $z_{k+1}, ..., z_K$. The neurons $\nu_{k+1}, ..., \nu_K$ just need to be clamped to a strong positive (negative) current to represent $z_j = 1$ ($z_j = 0$).
Buesing et al. [2011] present also a continuous version of neural sampling by analyzing the discrete sampling network in the limit $dt \to 0$. A continuous version allows for updating all neurons in parallel.

Implementation of Boltzmann Machines: For the particular case of Boltzmann distributions, the NCC (see Equation 2.18) is satisfied by neurons ν_k with the membrane potential

$$u_k(t) = b_k + \sum_{i=1}^{K} W_{ki} z_i(t) \ , \tag{2.22}$$

with the bias b_k of neuron ν_k and the weight W_{ki} of the connection from neuron ν_i to neuron ν_k. $W_{ki} z_i(t)$ is the shape of the Postsynaptic Potential (PSP) in the course of the membrane potential of the neuron ν_k caused by the firing of neuron ν_i with a square pulse. Equation 2.22 implies that all neurons have the synaptic time constant $\tau_{\text{syn}} = \tau_{\text{ref}} = \tau$.
For instance, if we have some bivariate probability distribution $p(z_1, z_2)$ over binary RVs Z_1 and Z_2, the biases of the neurons ν_1 and ν_2 are implemented as (see Equations 2.18

and 2.22)

$$b_1 = \log \frac{p\left(z_1 = 1, z_2 = 0\right)}{p\left(z_1 = 0, z_2 = 0\right)} \text{ and } b_2 = \log \frac{p\left(z_1 = 0, z_2 = 1\right)}{p\left(z_1 = 0, z_2 = 0\right)} \,, \tag{2.23}$$

while the weights between ν_1 and ν_2 are both

$$W_{12} = W_{21} = \log \frac{p\left(z_1 = 0, z_2 = 0\right) p\left(z_1 = 1, z_2 = 1\right)}{p\left(z_1 = 1, z_2 = 0\right) p\left(z_1 = 0, z_2 = 1\right)} \,. \tag{2.24}$$

Figure 2.6 illustrates a possible spike pattern of two stochastic neurons ν_1 and ν_2 which are sampling from a probability distribution $p(z_1, z_2)$. Both neurons have an absolute refractory time τ after firing.
BMs provide an adequate model for many real-world inference tasks like e.g. the binocular rivalry described by *Alais and Blake* [2005]. However, BMs maximally allow second-order probability distributions (see Equation 2.8), but numerous real-world phenomena require probabilistic models with higher-order dependencies between the RVs. *Pecevski et al.* [2011] introduce five methods which allow to extend the theory of neural sampling to higher-order probability distributions. In the course of this thesis, two of these approaches are applied, which are outlined in the following.

Implementation 1 of Bayesian Networks: Implementation 1 from *Pecevski et al.* [2011] exploits the fact that *any* probability distribution p can be reduced to a Boltzmann distribution [*Ackley et al.*, 1985]. In particular, a higher-order probability distribution $p\left(z_1, ..., z_K\right)$ over K binary RVs $Z_1, ..., Z_K$ can be reduced to a Boltzmann distribution in the following way: For each n^{th}-order factor with $n > 2$, 2^n auxiliary binary RVs $X_1, ..., X_{2^n}$ are introduced, such that the target probability distribution $p\left(\mathbf{z}\right)$ can be represented as marginal distribution

$$p\left(\mathbf{z}\right) = \sum_{\mathbf{x} \in \mathscr{X}} p\left(\mathbf{z}, \mathbf{x}\right) \tag{2.25}$$

of the extended distribution $p\left(\mathbf{z}, \mathbf{x}\right)$. Here, $\mathscr{X}$ denotes the set of all possible assignments of the random vector $\mathbf{X}$. The auxiliary variables are chosen such that each possible assignment of the higher-order factor is covered.
Let us consider the concrete example

$$p\left(z_1, z_2, z_3\right) = p\left(z_1\right) p\left(z_2\right) p\left(z_3 | z_1, z_2\right) \,. \tag{2.26}$$

The probability distribution contains the third-order factor $p\left(z_3 | z_1, z_2\right)$. The reduction of $p\left(z_1, z_2, z_3\right)$ to a Boltzmann distribution will involve 8 additional auxiliary RVs $X_{000}, X_{001}, ..., X_{111}$, one for each possible assignment of $p\left(z_3 | z_1, z_2\right)$. For example, the auxiliary RV X_{001} assumes the value 1 only if $z_1 = 0$, $z_2 = 0$ and $z_3 = 1$. For all other assignments, the variable remains 0.
In the neuron model, this can be achieved by setting the connection strengths to $M_{\text{exc}} = \alpha$ between the neurons corresponding to the variables Z_3 and X_{001} and to $M_{\text{inh}} = -\alpha$

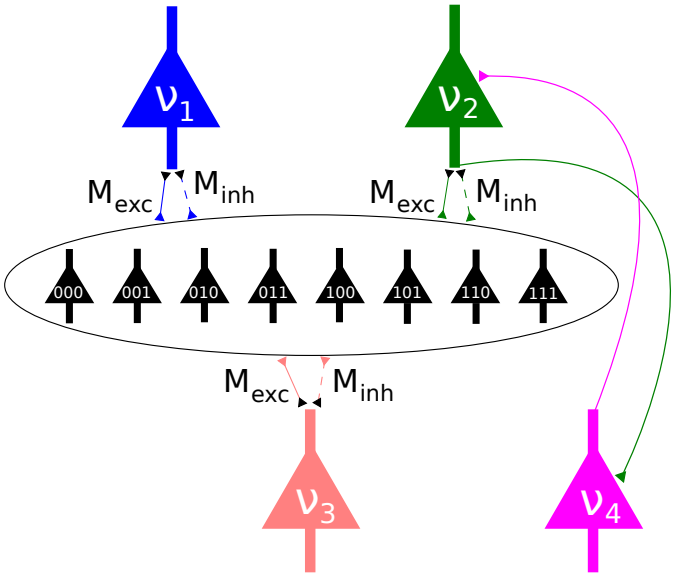


Figure 2.7: Implementation 1 for the visual perception experiment in Figure 2.2. There are eight auxiliary neurons (*black*), one for each possible assignment of the third-order factor $p(z_3|z_1, z_2)$. The auxiliary neurons determine the activity of the neurons ν_1, ν_2 and ν_3: there are strong excitatory connections M_{exc} between the auxiliary neuron and ν_i if the i-th digit of the auxiliary neuron is a 1, and strong inhibitory connections M_{inh} otherwise. The second-order factor $p(z_4|z_2)$ is represented via direct connections between the neurons ν_2 and ν_4.

between the neurons corresponding to the variables Z_1, Z_2 and X_{001}. *Pecevski et al.* [2011] specify the connection strength α as

$$\alpha = 10 \cdot \max\left(p\left(z_3|z_1, z_2\right)\right) \ . \tag{2.27}$$

This value arises from the fact that a neuron with a bias of α will always be triggered to elicit a spike irrespective of the states of the remaining RVs. By analogy, a neuron with a bias of $-\alpha$ will always remain silent regardless of the input from the other neurons.
The individual values of the factor $p(z_3|z_1, z_2)$ are introduced through the bias of the neurons which correspond to the auxiliary RVs. For instance, the bias of the neuron corresponding to the RV X_{001} is calculated via

$$b_{001} = \log\left(\mu \frac{p\left(z_3 = 1|z_1 = 0, z_2 = 0\right)}{\min\left[p\left(z_3|z_1, z_2\right)\right]} - 1\right) - \underbrace{\eta\left(z_1 = 0, z_2 = 0, z_3 = 1\right)}_{=1} M_{\text{exc}} \ , \tag{2.28}$$

where μ is an arbitrary factor ensuring that the argument of the logarithm stays larger than 0 for all possible assignments z_1, z_2 and z_3. The function $\eta(\mathbf{v})$ returns the L^1-norm of $\mathbf{v}$. The first term of Equation 2.28 contains the NCC from Equation 2.18: It connects the conditional probability associated with the auxiliary RVs to the membrane potential of the corresponding neuron. The second term provides a strong inhibition of the neuron

to ensure that the neuron may only elicit a spike if the assignments of the RVs Z_1, Z_2 and Z_3 assume the appropriate combination.
If, for example, the assignments of the random vector $\mathbf{Z}$ are $z_1 = 1$, $z_2 = 0$ and $z_3 = 1$, which is not compatible with the combination 001, then the voltage of the neuron corresponding to the auxiliary RV X_{001} is forced to about $-\alpha$ by the input of ν_1 and the neuron remains silent. Another example is the assignment $z_1 = 1$, $z_2 = 0$ and $z_3 = 0$, which is not compatible with the combination 101. In this case, the neuron which corresponds to the RV X_{101} gets not enough input and thus remains at the low membrane potential of about $-\alpha$.
Figure 2.7 illustrates how the visual perception experiment from Figure 2.2 is modeled via Implementation 1. The neural network contains the four neurons $\nu_1, ..., \nu_4$, which represent the RVs $Z_1, ..., Z_4$, and additionally 8 auxiliary neurons representing the RVs $X_{000}, X_{001}, ..., X_{111}$, one for each possible assignment of the factor $p\,(z_3|z_1, z_2)$.
According to *Levin et al.* [2006], the convergence of the sampling distribution towards the target probability distribution $p\,(\mathbf{z})$ is very slow due to the introduction of additional RVs X. The following paragraph will present another implementation of BNs with spiking neurons in which the original RVs $Z_1, ..., Z_K$ directly fulfill the NCC without the need of additional RVs.

Implementation 2 of Bayesian Networks: Implementation 2 from *Pecevski et al.* [2011] provides the ability to sample from higher-order probability distributions over the binary random vector $\mathbf{Z}$ through a Markov blanket expansion of the NCC (see Equation 2.18). The *Markov blanket* B_k of a node Z_k in a Bayesian network is defined as the set of all parents, children and co-parents of this node [*Pearl*, 1988]. By definition, it has the property that, once any assignment $\mathbf{v}$ to the RVs $\mathbf{Z}^{B_k}$ in the Markov blanket has been fixed, Z_k is independent from all other RVs in the graph:

$$p(z_k|\mathbf{z}_{\setminus k}) = p(z_k|\mathbf{z}^{B_k})\ . \tag{2.29}$$

The Markov blanket B_k "shields" the RV Z_k from the rest of the nodes [*Bishop*, 2006]. For instance, the Markov blanket of node Z_1 in Figure 2.2 consists of its co-parent Z_2 and its child Z_3. By fixing the RVs Z_2 and Z_3, Z_1 becomes conditionally independent of the RV Z_4.
The NCC from Equation 2.18 can then be expanded as

$$\log \frac{p(z_k(t) = 1|\mathbf{z}^{B_k}(t))}{p(z_k(t) = 0|\mathbf{z}^{B_k}(t))} = \sum_{\mathbf{v} \in Z^{B_k}} \underbrace{\log \frac{p(z_k = 1|\mathbf{z}^{B_k} = \mathbf{v})}{p(z_k = 0|\mathbf{z}^{B_k} = \mathbf{v})}}_{w_k^{\mathbf{v}}} \cdot [\mathbf{z}^{B_k}(t) = \mathbf{v}]\ , \tag{2.30}$$

where $\mathbf{v}$ runs through all possible assignments $\mathbf{z}^{B_k}$. The expression $[\mathbf{z}^{B_k}(t) = \mathbf{v}]$ is 1 if the condition inside the brackets is true, otherwise 0. Equation 2.30 implies that for satisfying the NCC it suffices if there are $2^{|B_k|}$ *auxiliary neurons*, one for each possible assignment $\mathbf{v}$, that become active if and only if the RVs $\mathbf{Z}^{B_k}$ assume $\mathbf{v}$. The current values $\mathbf{z}^{B_k}$ of the RVs $\mathbf{Z}^{B_k}$ are encoded in the firing activity of their associated *principal neurons* ν_k.

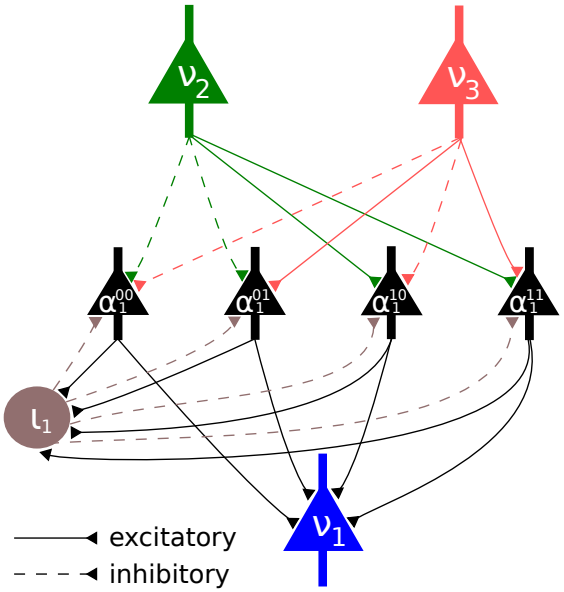


Figure 2.8: Implementation 2 for the example in Equation 2.26. The figure is taken from *Pecevski et al.* [2011]. It shows the neural representation of the Markov blanket of the RV z_1. There are 4 auxiliary neurons, one for each possible assignment $\mathbf{v}$ to the RVs z_2 and z_3. The corresponding principal neurons ν_i connect to the auxiliary neuron $\alpha_1^{\mathbf{v}}$ with an excitatory (inhibitory) connection if $v_i = 1$ (0). The auxiliary neurons connect with strong (ideally ∞) excitatory synapses to both the principal neuron ν_1 and the inhibitory interneuron ι_1 causing them to fire immediately upon stimulation. The inhibitory neuron connects back to the auxiliary neurons with strong inhibitory weights. This ensures that all auxiliary neurons remain silent for τ_{ref} whenever one of them has spiked.

The NCC is satisfied in the neural implementation by choosing appropriate values for the excitability of the auxiliary neurons and a specific connectivity pattern between the principal and the auxiliary neurons.
Figure 2.8 shows the neural implementation of the Markov blanket of variable z_1 for the concrete example distribution in Equation 2.26. It consists of the variables z_2 and z_3. The associated principal neurons ν_2 and ν_3 thus connect directly to the auxiliary neurons $\alpha_1^{\mathbf{v}}$. A connection from the i^{th} principal neuron of the Markov blanket to $\alpha_1^{\mathbf{v}}$ is excitatory (inhibitory) if the assignment $\mathbf{v}$ contains a 1 (0) a position i. At each moment in time only the auxiliary neuron $\alpha_1^{\mathbf{v}}$ corresponding to the current state of the inputs $\mathbf{z}^{B_k}(t) = \mathbf{v}$ can fire (with a probability determined by the NCC), but only if it is not laterally inhibited due to a recent spike from another auxiliary neuron. All auxiliary neurons $\alpha_k^{\mathbf{v}}$ connect with strong excitatory synapses to both the principal neuron ν_k and the local inhibitory interneuron ι_k, causing all efferent neurons to fire whenever they fire. The local inhibitory interneuron ι_k connects back to the auxiliary neurons with strong inhibitory synapses. This ensures that all auxiliary neurons remain silent for a time $\tau = \tau_{\text{ref}} = \tau_{\text{syn}}$ whenever

one of them has spiked.
The term $w_k^{\mathbf{v}}$ from Equation 2.30 is implemented via the bias $b_k^{\mathbf{v}}$ of the auxiliary neuron that corresponds to the assignment $\mathbf{v}$, which ensures the satisfaction of the NCC:

$$b_k^{\mathbf{v}} = \log \frac{p(z_k = 1 | \mathbf{z}^{B_k} = \mathbf{v})}{p(z_k = 0 | \mathbf{z}^{B_k} = \mathbf{v})} - \eta(\mathbf{v}) M_k^{\mathbf{v}} \ . \tag{2.31}$$

The factor $\eta(\mathbf{v})$ denotes the L^1-norm of the vector $\mathbf{v}$ and $M_k^{\mathbf{v}}$ represents the excitatory synaptic weight from the principal neuron ν_i to the auxiliary neuron $\alpha_k^{\mathbf{v}}$. In the case $\mathbf{v} \neq \mathbf{z}^{B_k}(t)$, the neuron $\alpha_{\mathbf{v}}^k$ remains silent either due to insufficient input or due to the strong inhibitory connections from the principal neurons.
The excitatory synaptic weight $M_k^{\mathbf{v}}$ from the principal neuron ν_i to an auxiliary neuron $\alpha_k^{\mathbf{v}}$ is set to

$$M_k^{\mathbf{v}} = \max \left(\log \frac{p(z_k = 1 | \mathbf{z}^{B_k} = \mathbf{v})}{p(z_k = 0 | \mathbf{z}^{B_k} = \mathbf{v})} + 10,\ 0 \right) \ . \tag{2.32}$$

Similarly, the inhibitory synaptic weight $M_k^{\mathbf{v}}$ from both a principal neuron ν_i and the local inhibitory interneuron ι_k to an auxiliary neuron $\alpha_k^{\mathbf{v}}$ is set to

$$M_k^{\mathbf{v}} = \min \left(- \log \frac{p(z_k = 1 | \mathbf{z}^{B_k} = \mathbf{v})}{p(z_k = 0 | \mathbf{z}^{B_k} = \mathbf{v})} - 10,\ 0 \right) \ . \tag{2.33}$$

The biases of the principal neurons and the local inhibitory interneurons amount to $b = -10$. All efferent synaptic weights of the auxiliary neurons are set to $w = 30$ in order to ensure that the postsynaptic neurons fire upon each incoming spike.

2.1.6 Deterministic Neuron and Synapse Models

This section discusses the dynamics of deterministic neuron and synapse models. We start with the description of the membrane dynamics of the leaky integrate-and-fire (LIF) neuron and study the impact of synaptic input via current-based and conductance-based synapses on the course of its membrane potential. Thereafter, we discuss the membrane dynamics of the adaptive exponential integrate-and-fire (AdEx) neuron model, which is implemented on the BrainScaleS wafer-scale hardware system (see Section 2.2). The last paragraph describes the Tsodyks-Markram (TM) mechanism of synaptic plasticity which models the limitedness of synaptic neurotransmitters. The TM model is crucial for LIF sampling, which will follow in Section 2.1.7.

Leaky Integrate-and-Fire Neuron Model The membrane dynamics of the leaky integrate-and-fire (LIF) neuron can be described by the following differential equation [*Gerstner and Kistler*, 2002]:

$$C_{\mathrm{m}} \frac{\mathrm{d}V(t)}{\mathrm{d}t} + g_{\mathrm{l}} \left(V(t) - E_{\mathrm{l}} \right) + \sum_{\text{syn } i} I_i(t) + I_{\mathrm{ext}}(t) = 0 \ . \tag{2.34}$$

C_{m}, g_{l} and E_{l} represent the membrane capacitance, the membrane leakage conductance and the membrane leakage potential, respectively. The membrane time constant τ_{m}

is determined by g_l according to $\tau_\mathrm{m} = C_\mathrm{m}/g_\mathrm{l}$. $I_\mathrm{ext}(t)$ subsumes all external currents, while I_i is the synaptic input current due to recurrent connections in the network or diffusive background noise. Equation 2.34 describes the dynamics of what is called the *free membrane potential* of the LIF neuron.
If a certain threshold voltage V_th is exceeded, the neuron emits a spike and is reset to the reset voltage V_reset for the so-called *refractory time* τ_ref.
One can distinguish *current-based* and *conductance-based* synapse models. For exponential-decaying *current-based* synapses, I_i from Equation 2.34 is the synaptic input current

$$I_i(t) = w_i \sum_{\text{spike } k} \Theta\left(t - t_k\right) \cdot \exp\left(-\frac{t - t_k}{\tau_i}\right) \ . \tag{2.35}$$

w_i determines the height of the PSP and thus represents the synaptic weight. If w_i is positive (negative), the connection is excitatory (inhibitory). τ_i is the synaptic time constant, determining the decay speed of the synaptic current in Equation 2.35. The sum runs over the arrival times of incoming spikes via synapse i. Θ is the Heaviside step function. The synaptic current is fully determined by w_i and τ_i. We assume from now on that the synaptic time constant is identically $\tau_i = \tau_\mathrm{syn}$ for each synapse of the neuron.
With the choice of I_i in Equation 2.35, Equation 2.34 becomes a linear differential equation and can be solved analytically. *Bytschok* [2011] calculates the PSP time course $V_\mathrm{PSP}(t)$ for a current-based LIF neuron upon an incoming spike at time t_spike:

$$V_\mathrm{PSP}(t) = \frac{w_i}{g_\mathrm{l} \cdot \tau_\mathrm{m} \cdot \left(\frac{1}{\tau_\mathrm{syn}} - \frac{1}{\tau_\mathrm{m}}\right)} \cdot \left[\exp\left(-\frac{t - t_\mathrm{spike}}{\tau_\mathrm{m}}\right) - \exp\left(-\frac{t - t_\mathrm{spike}}{\tau_\mathrm{syn}}\right)\right] \ . \tag{2.36}$$

For *conductance-based* synapses, the current I_i from Equation 2.34 becomes

$$I_i(t) = g_i(t)\left(V(t) - E_i^\mathrm{rev}\right) \ . \tag{2.37}$$

The quantities g_i and E_i^rev are the synaptic conductance and the synaptic reversal potential, respectively. The synaptic conductance g_i amounts to

$$g_i(t) = w_i \sum_{\text{spike } k} \Theta\left(t - t_k\right) \cdot \exp\left(-\frac{t - t_k}{\tau_\mathrm{syn}}\right) \ , \tag{2.38}$$

by analogy to Equation 2.35. In difference to the current-based model, each synapse comes with its individual reversal potential E_i^rev. The amplitude of the injected current upon arrival of a spike depends on the value of the membrane voltage itself, which makes the system nonlinear. Equation 2.34 then yields

$$C_\mathrm{m}\frac{\mathrm{d}V(t)}{\mathrm{d}t} = -g_\mathrm{l}\left(V(t) - E_\mathrm{l}\right) - \sum_{\text{syn } i} g_i(t)\left(V(t) - E_i^\mathrm{rev}\right) - I_\mathrm{ext}(t) \ , \tag{2.39}$$

Due to the nonlinearity in $V(t)$, Equation 2.39 cannot be solved analytically and a closed-form solution of the voltage course V_PSP of a PSP cannot be calculated. However, an approximative solution of Equation 2.39 can be obtained for the case that the neuron is exposed to a spiking noisy environment, which will be presented in Section 2.1.7.

Capacity of the membrane	C_m
Leakage conductance	g_l
Synaptic conductance	g_i
Leakage potential	E_l
Reversal potential	E_i^rev
Slope factor	Δ_th
Effective threshold potential	V_th
Reset potential	V_reset
Adaptation time constant	τ_w
Adaptation coupling parameter	a
Spike-triggered adaptation	b

Table 2.1: Parameters of the adaptive exponential integrate-and-fire neuron model

Adaptive Exponential Integrate-and-Fire Neuron Model The membrane dynamics of the neurons integrated on the BrainScaleS wafer-scale hardware system (see Section 2.2) follow those of the adaptive exponential integrate-and-fire (AdEx) neuron model with conductance-based synapses [*Brette and Gerstner*, 2005]. The model is described by two differential equations. The first differential equation determines the time course of the membrane potential $V(t)$:

$$\begin{aligned} -C_\mathrm{m}\frac{\mathrm{d}V(t)}{\mathrm{d}t} &= g_\mathrm{l}\left(V(t)-E_\mathrm{l}\right) - g_\mathrm{l}\,\Delta_\mathrm{th}\exp\left(\frac{V(t)-V_\mathrm{th}}{\Delta_\mathrm{th}}\right) \\ &+ \sum_{\mathrm{syn}\ i} g_i(t)\left(V(t)-E_i^\mathrm{rev}\right) + w(t)\ . \end{aligned} \tag{2.40}$$

The second differential equation describes the temporal progress of the so-called *adaptation current* $w(t)$:

$$-\tau_\mathrm{w}\frac{\mathrm{d}w(t)}{\mathrm{d}t} = w(t) - a(V(t)-E_\mathrm{l})\ . \tag{2.41}$$

Table 2.1 lists the parameters of the two differential equations.
The process of spike generation is embedded in the exponential term: A spike occurs each time the membrane potential $V(t)$ grows towards infinity. For practical reasons, the integration of the AdEx model equations usually stops at the time at which the membrane potential crosses some threshold Θ. Exactly this time is registered as spike time. Upon each elicited spike, the membrane potential and the adaptation current are abruptly reset, which serves as a simplification of the downswing of the action potential:

$$V \rightarrow V_\mathrm{reset} \tag{2.42}$$

$$w \rightarrow w + b\ . \tag{2.43}$$

The AdEx model can be reduced to the LIF model by taking the limit $\Delta_\mathrm{th} \rightarrow 0$ and setting $w = 0$.

Tsodyks-Markram Mechanism The Tsodyks-Markram (TM) mechanism of synaptic plasticity, or Short-Term Plasticity (STP), models the limitedness of neurotransmitters in a synapse [*Tsodyks and Markram*, 1997; *Markram et al.*, 1998]. It takes account of the fact that the neurotransmitters which are released upon an action potential are not instantaneously available for a subsequent action potential and first need to be recovered. Therefore, the actual *effective* synaptic strength E is equal to the *absolute* synaptic strength A, which is the maximum response to a presynaptic spike, only if all neurotransmitters are on the spot and otherwise smaller. In the following, the dynamics of the TM mechanism are presented.
Each presynaptic action potential of a neuron uses some fraction of the absolute synaptic strength A, which is defined as the *utilization of synaptic efficacy parameter* U. The variable U changes for each spike. The running variable of U is referred to as u and is determined by the equation

$$u_{n+1} = U_0 + u_n \left(1 - U_0\right) \exp\left(-\frac{\Delta t}{\tau_{\text{facil}}}\right) , \tag{2.44}$$

where Δt is the time interval between the n^{th} and the $(n+1)^{\text{th}}$ spike. Without an arriving action potential, u decays exponentially with the time constant τ_{facil} towards its resting value U_0.
The fraction of available synaptic strength is described by the *recovered synaptic efficacy parameter* R which follows the update rule

$$R_{n+1} = R_n \left(1 - u_{n+1}\right) \exp\left(-\frac{\Delta t}{\tau_{\text{rec}}}\right) + 1 - \exp\left(-\frac{\Delta t}{\tau_{\text{rec}}}\right) , \tag{2.45}$$

with the *recovery time constant* τ_{rec}. In the absence of action potentials, the R increases towards 1. The EPSP generated by any action potential then amounts to

$$\text{EPSP}_n = A \cdot R_n \cdot u_n . \tag{2.46}$$

Inhibitory synaptic connections are characterized by $\tau_{\text{facil}} \to 0$ which yields $u_n = U_0$.

2.1.7 LIF Sampling

This section illustrates the theoretical background of LIF sampling, the sampling from probability distributions with recurrent networks of deterministic leaky integrate-and-fire (LIF) neurons. First the characteristics of the membrane dynamics of a LIF neuron in the so-called High-Conductance-State (HCS) are described. In the HCS, the free membrane potential of a LIF neuron can be described by an Ornstein-Uhlenbeck (OU) process, which is outlined thereafter. This mathematical prerequisite finally allows the neuron to closely reproduce a logistic activation function, which connects the mean membrane potential of the neuron to the probability to find it in the refractory state. This analogy to the NCC from Equation 2.18 is the basis for performing neural sampling with LIF neurons.

The Leaky Integrate-and-Fire Neuron in a Spiking Noisy Environment Due to the nonlinearity in the membrane potential of a LIF neuron with conductance-based synapses (see Equation 2.37), Equation 2.39 cannot be solved analytically and a closed-form solution of the voltage course V_{PSP} of a PSP cannot be calculated. An approximative solution of Equation 2.39, however, can be obtained for the case that the neuron is exposed to a spiking noisy environment, which will be outlined in the following.
First we divide both sides of Equation 2.39 by the total conductance $g_{\mathrm{tot}}(t) = g_l + \sum_i g_i(t)$ and define the effective membrane potential V_{eff} and the effective time constant τ_{eff} as

$$V_{\mathrm{eff}}(t) = \frac{g_{\mathrm{l}} E_{\mathrm{l}} + \sum_i g_i(t) \cdot E_i^{\mathrm{rev}} + I_{\mathrm{ext}}}{g_{\mathrm{tot}}(t)} \quad \text{and} \quad \tau_{\mathrm{eff}}(t) = \frac{C_{\mathrm{m}}}{g_{\mathrm{tot}}(t)} \; . \tag{2.47}$$

Equation 2.39 can then be written as

$$\tau_{\mathrm{eff}}(t) \frac{\mathrm{d}V(t)}{\mathrm{d}t} = V_{\mathrm{eff}}(t) - V(t) \; . \tag{2.48}$$

A spiking noisy environment can be generated via the stimulation by excitatory and inhibitory Poisson sources of rate $\nu_i \to \infty$ and synaptic weights $w_i \to 0$ of the connections from the Poisson sources to the neuron. The neuron then enters the so-called high-conductance state (HCS) which can be characterized by an approximately constant total average conductance $\overline{g_{\mathrm{tot}}(t)}$, which is larger than the leakage conductance g_{l}, and thus a τ_{eff} smaller than τ_{m} [*Destexhe et al.*, 2003]. *Bytschok* [2011] calculates the mean $\overline{V(t)}$ and the variance $\sigma^2_{V(t)}$ of the free membrane potential of a LIF neuron in the HCS:

$$\overline{V(t)} = \overline{V_{\mathrm{eff}}(t)} = \frac{g_{\mathrm{l}} E_{\mathrm{l}} + \sum_i \overline{g_i(t)} E_i^{\mathrm{rev}} + I_{\mathrm{ext}}}{\overline{g_{\mathrm{tot}}(t)}} \tag{2.49}$$

and

$$\sigma^2_{V(t)} = \sum_i \nu_i S_i^2 \left(\frac{\tau_{\mathrm{eff}}}{2} + \frac{\tau_{\mathrm{syn},i}}{2} - 2 \frac{\tau_{\mathrm{eff}} \tau_{\mathrm{syn},i}}{\tau_{\mathrm{eff}} + \tau_{\mathrm{syn},i}} \right) \tag{2.50}$$

with

$$S_i = \frac{w_i \left(E_i^{\mathrm{rev}} - \overline{V(t)} \right)}{C_{\mathrm{m}} \left(\frac{1}{\tau_{\mathrm{eff}}} - \frac{1}{\tau_{\mathrm{syn},i}} \right)} \; . \tag{2.51}$$

With $\tau_{\mathrm{eff}} \to 0$, which implies that the membrane potential $V(t)$ instantly follows the effective membrane potential $V_{\mathrm{eff}}(t)$, Equations 2.47 and 2.48 can be rewritten as

$$V(t) \approx V_{\mathrm{eff}}(t) = \frac{g_{\mathrm{l}} E_{\mathrm{l}} + \sum_i \overline{g_i(t)} E_i^{\mathrm{rev}} + \sum_i \Delta g_i(t) E_i^{\mathrm{rev}} + I_{\mathrm{ext}}}{\overline{g_{\mathrm{tot}}(t)} + \sum_i \Delta g_i(t)} \tag{2.52}$$

with the fluctuations of the synaptic conductances

$$\Delta g_i(t) = g_i(t) - \overline{g_i(t)} \; . \tag{2.53}$$

The mean and the variance of the conductance evoked by the stimulation via a single Poisson source with the rate ν_i, connected to the neuron by a synapse with weight w_i and time constant τ_{syn}, can be calculated analytically [*Bytschok*, 2011]:

$$\overline{g_i(t)} = w_i \nu_i \tau_{\text{syn}} \quad \text{and} \quad \sigma^2_{g_i(t)} = \frac{w_i^2 \nu_i \tau_{\text{syn}}}{2} \; . \tag{2.54}$$

The relative fluctuations of the synaptic conductance can be described by the coefficient of variation c_{V}:

$$c_{\text{V}} = \frac{\sigma_{g_i(t)}}{\overline{g_i(t)}} = \sqrt{\frac{1}{2\nu_i \tau_{\text{syn}}}} \; . \tag{2.55}$$

In the limit $\nu_i \to \infty$, the fluctuations of the synaptic conductances become small, which allows an expansion of Equation 2.52 in Δg_i. The first-order approximation amounts to [*Petrovici et al.*, 2013]:

$$V(t) = \frac{g_{\text{l}} E_{\text{l}} + \sum_i g_i(t) E_i^{\text{rev}} + I_{\text{ext}}}{\overline{g_{\text{tot}}(t)}} \; , \tag{2.56}$$

which reveals that in the limit of high input frequencies the free membrane potential (i.e. $V_{\text{th}} \to \infty$) of a LIF neuron depends linearly on the synaptic input $\sum_i g_i(t) E_i^{\text{rev}}$.

The free membrane potential in the HCS as an Ornstein-Uhlenbeck process The description of the free membrane potential of a LIF neuron in the HCS as an Ornstein-Uhlenbeck process is the mathematical prerequisite which allows the application of the neural sampling theory (see Section 2.1.5) with LIF neurons.
An *Ornstein-Uhlenbeck* (OU) process [*Uhlenbeck and Ornstein*, 1930] is a stochastic process defined by the ODE

$$\mathrm{d}x(t) = \Theta \cdot (\mu - x(t)) + \sigma \cdot \mathrm{d}W(t) \; . \tag{2.57}$$

Here, Θ is a positive constant and $W(t)$ a stationary delta-correlated random process with mean 0 and variance σ^2, a so-called *Wiener process*. Equation 2.57 can intuitively be understood as the Brownian motion of the step size $\sigma \mathrm{d}W(t)$ of a particle with an attractor at μ towards which it decays exponentially with the time constant Θ. *Ricciardi* [1977], among others, proves that the probability density function (PDF) of the OU process

$$f(x, t|x_0) = \sqrt{\frac{\Theta}{\pi\sigma^2 \left(1 - \mathrm{e}^{-2\Theta t}\right)}} \exp\left(-\frac{\Theta}{\sigma^2} \left[\frac{\left(x - x_0 \mathrm{e}^{-\Theta t}\right)^2 - \mu}{1 - \mathrm{e}^{-2\Theta t}} \right] \right) \tag{2.58}$$

is the unique solution of the *Fokker-Planck equation*

$$\frac{\partial f(x,t)}{\partial t} = \Theta \cdot \frac{\partial}{\partial x} \left[(x - \mu) f \right] + \frac{\sigma^2}{2} \frac{\partial^2 f(x,t)}{\partial x^2} \tag{2.59}$$

which satisfies the initial condition

$$\lim_{t \to 0} f(x, t|x_0) = \delta \left(x - x_0 \right) \; . \tag{2.60}$$

Gerstner and Kistler [2002] and *Petrovici et al.* [2013] show that the distribution $f(\xi, t)$ of the synaptic noise $\xi := \xi(t) = \sum_i g_i(t) E_i^{\text{rev}}$ of a LIF neuron with conductance-based synapses obeys the Fokker-Planck equation:

$$\frac{\partial f(\xi,t)}{\partial t} = \frac{1}{\tau_{\text{syn}}} \frac{\partial}{\partial \xi} \left[\left(\xi - \sum_i \nu_i \Delta\xi_i \tau_{\text{syn}} \right) \cdot f(\xi,t) \right] + \frac{\sum_i \nu_i \Delta\xi_i^2 \tau_{\text{syn}}}{2\tau_{\text{syn}}} \frac{\partial^2 f(\xi,t)}{\partial \xi^2} \,. \tag{2.61}$$

This fact implies that the temporal evolution of the synaptic noise of a LIF neuron can be described by an OU process.
Due to Equation 2.56, the free membrane potential of a LIF neuron in the HCS depends linearly on the synaptic noise $\xi(t)$ in the limit $\tau_{\text{eff}} \to 0$. In this situation the free membrane potential of the a LIF neuron can therefore also be approximated by an OU process. The parameters of the OU process which describes the free membrane potential can be deduced from Equations 2.49 and 2.50 with $\tau_{\text{eff}} \to 0$:

$$\Theta = \frac{1}{\tau_{\text{syn}}} \tag{2.62}$$

$$\mu = \frac{g_{\text{l}} E_{\text{l}} + \sum_i \nu_i w_i E_i^{\text{rev}} \tau_{\text{syn}} + I_{\text{ext}}}{\overline{g_{\text{tot}}(t)}} \tag{2.63}$$

$$\sigma^2 = \frac{\sum_i \nu_i \left(w_i \left(E_i^{\text{rev}} - \mu \right) \right)^2 \tau_{\text{syn}}}{\overline{g_{\text{tot}}(t)}} \,. \tag{2.64}$$

In the next section, we will derive the activation function of a LIF neuron in the HCS, which connects the mean membrane potential of the neuron with the probability to find it in the refractory state. This finally allows to translate the biases and weights of the stochastic neurons from the abstract model described in Section 2.1.5 to the LIF domain.

Neural Sampling via Leaky Integrate-and-Fire Neurons with Conductance-Based Synapses *Petrovici et al.* [2013] show that a single LIF neuron in the HCS, whose membrane dynamics thus can be described by an OU process, can closely reproduce the activation function $p(z_k = 1)$ (see Equation 2.19). The activation function $p(z_k = 1)$ creates a link between the mean membrane potential of a neuron ν_k and the probability to find it in the refractory state $z_k = 1$. In the following, the shape of the activation function of a LIF neuron is presented. Finally, with the help of the activation function, the parameters of the abstract neuron model from Equation 2.22 can be directly translated to the parameters of LIF neurons.
By analogy with the abstract neuron model of Section 2.1.5, the RV z_k is 1 if neuron ν_k is refractory. The state of the LIF neuron can be divided into two "modes": A "bursting mode" in which the neuron emits spikes sequentially with an inter-spike interval (ISI) of $\Delta t = \tau_{\text{ref}}$ and a "freely evolving mode" between these bursts, during which the membrane potential evolves freely in the subthreshold regime. Figure 2.9 shows a typical course of the membrane potential of a LIF neuron in the HCS with its "bursting states" (gray boxes) and "freely evolving states" (white area).

With these two "modes", the activation function of a LIF neuron assumes the following shape:

$$p\left(z_k = 1\right) = \frac{\sum_n P_n \cdot n \cdot \tau_{\text{ref}}}{\sum_n P_n \cdot \left(n \cdot \tau_{\text{ref}} + T_n\right)} \,. \tag{2.65}$$

Here, P_n is the probability of the occurrence of an n-spike-burst, lasting for $n\tau_{\text{ref}}$. T_n is the average time interval between the end of the previous n-spike-burst and the next spike. Summing over the contributions of all possible burst lengths, the nominator in Equation 2.65 represents the mean bursting time of the neuron, while the denominator is the sum of both, the mean bursting time and the mean free evolving time, equaling the total propagation time of the LIF neuron.
As we have seen in the previous paragraph, the free membrane potential of a neuron in the HCS can be described by an OU process. Thus P_n and T_n can be calculated iteratively making use of the PDF of the OU process (see Equation 2.58). The pink areas in Figure 2.9 depict the PDFs of the membrane potential of a LIF neuron in a HCS after it leaves the refractory state.
We will in the following denote the spike times of within an n-spike-burst by $t_0, ..., t_{n-1}$ and define $V_i := V(t_i)$. For an arbitrary burst length n we can then write:

$$\begin{aligned} P_n &= p\left(V_n < V_{\text{th}}, V_{n-1} \geq V_{\text{th}}, ..., V_1 \geq V_{\text{th}} | V_0 = V_{\text{th}}\right) \\ &= \left(1 - \sum_{i=1}^{n-1} P_n\right) \int_{V_{\text{th}}}^{\infty} \mathrm{d}V_{n-1}\, p\left(V_{n-1} | V_{n-1} \geq V_{\text{th}}\right) \left[\int_{-\infty}^{V_{\text{th}}} \mathrm{d}V_n\, p\left(V_n | V_{n-1}\right)\right] \\ T_n &= \int_{V_{\text{th}}}^{\infty} \mathrm{d}V_{n-1}\, p\left(V_{n-1} | V_{n-1} \geq V_{\text{th}}\right) \left[\int_{-\infty}^{V_{\text{th}}} \mathrm{d}V_n\, p\left(V_n | V_n < V_{\text{th}}, V_{n-1}\right) \overline{T\left(V_n, V_{\text{th}}\right)}\right] \,, \end{aligned} \tag{2.66}$$

where $p(V_i|V_{i-1}) = f(V, \tau_{\text{ref}}|V_i)$ from Equation 2.58 and $p(V_i|\mathfrak{R}(V_i))$ is the renormalization of the PDF of V_i for all values that fulfill the condition $\mathfrak{R}(V_i)$ (dark pink areas in Figure 2.9). For example, the probability of a burst of length $n = 1$, which is the probability that the neuron enters the "freely evolving mode" after a spike, yields according to Equation 2.66:

$$P_1 = p\left(V_1 < V_{\text{th}} | V_0 = V_{\text{th}}\right) = \int_{V_{\text{th}}}^{\infty} \mathrm{d}V_0\, p\left(V_0 | V_0 \geq V_{\text{th}}\right) \left[\int_{-\infty}^{V_{\text{th}}} \mathrm{d}V_1\, p\left(V_1 | V_0\right)\right] \,. \tag{2.67}$$

By analogy, the average interval between the previous "burst" of length $n = 1$ and the next spike amounts to:

$$T_1 = \int_{V_{\text{th}}}^{\infty} \mathrm{d}V_0\, p\left(V_0 | V_0 \geq V_{\text{th}}\right) \left[\int_{-\infty}^{V_{\text{th}}} \mathrm{d}V_1\, p\left(V_1 | V_1 < V_{\text{th}}, V_0\right) \overline{T\left(V_1, V_{\text{th}}\right)}\right] \,. \tag{2.68}$$

$\overline{T(V_i, V_{\text{th}})}$ from Equation 2.66 denotes the mean first passage time (FPT) which represents the duration of reaching the threshold V_{th} for the first time when starting from V_i. If we assume a nonzero effective time constant $\tau_{\text{eff}} \ll \tau_{\text{syn}}$, we can find a first-order correction to the average FPT using an expansion in $\sqrt{\tau_{\text{eff}}/\tau_{\text{syn}}}$, according to *Brunel and Sergi*

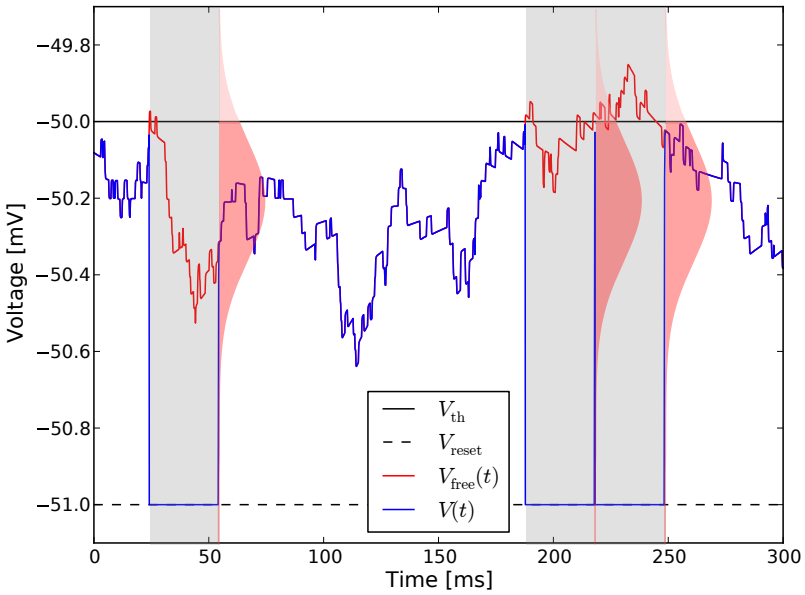


Figure 2.9: The *blue* curve shows a typical course of the membrane potential $V(t)$ of a LIF neuron in the HCS, while the *red* curve denotes its effective membrane potential $V_{\text{eff}}(t)$. Each time the membrane potential crosses the threshold voltage V_{th}, the neuron is set refractory (*gray box*) and its membrane potential is fixed to the reset potential V_{reset}. The *gray boxes* represent the "bursting mode" of the neuron, while the *white area* denotes the "free evolving mode". After the refractory period, the neuron is instantaneously pulled towards $V_{\text{eff}}(t)$ due to the small τ_{eff}. The *pink* area hereby represents the predicted probability distribution of $V_{\text{eff}}(t)$. The renormalized *dark pink* subthreshold area is used in Equation 2.66. The figure is drawn with inspiration from *Petrovici et al.* [2013].

[1998]:

$$\overline{T(V_0, V_{\text{th}})} = \tau_{\text{syn}}\sqrt{\pi}\int_{\frac{V_0-\mu}{\sigma}}^{\frac{V_{\text{th,eff}}-\mu}{\sigma}} \mathrm{d}x \exp\left(x^2\right)\cdot\left(\operatorname{erf}(x)+1\right) \ , \tag{2.69}$$

with the initial membrane potential V_0 and $V_{\text{th,eff}} \approx V_{\text{th}} - \zeta(0.5)\sqrt{\tau_{\text{eff}}/(2\tau_{\text{syn}})}$. Here, ζ denotes the Riemann Zeta function. The values μ and σ are taken from Equations 2.63 and 2.64.

Petrovici et al. [2013] show that the predicted activation function from Equation 2.65 closely reproduces the desired sigmoidal shape which is essential to perform neural sampling with LIF neurons.

The translation of membrane potentials from the LIF domain to the domain of abstract

stochastic neurons from Section 2.1.5 is provided by

$$u = \frac{\overline{V(t)} - \overline{V_k}^{b=0}}{\alpha} \,, \tag{2.70}$$

where $\overline{V_k}^{b=0}$ represents the value of $\overline{V_k}$ for which the probability $p(z_k = 1)$ to find the neuron in the refractory state is $\frac{1}{2}$. The factor α denotes the scaling factor between the two domains and can be deduced by fitting a logistic function to $p(z_k = 1)$ in Equation 2.65 and determining its slope.
The linear transformation of the membrane potentials in Equation 2.70 allows to directly specify the parameter translation of the bias b_k of neuron ν_k and the weight W_{ki} from neuron ν_i to ν_k from the abstract and the LIF domain. The average free membrane potential $\overline{V_k}^{b}$ of the LIF neuron ν_k that fulfills $p\left(z_k = 1\right) = \sigma\left(b_k\right)$ (see Equation 2.19) amounts to

$$\overline{V_k}^{b} = \alpha b_k + \overline{V_k}^{b=0} \,. \tag{2.71}$$

The translation of synaptic weights W_{ki} from the domain of abstract stochastic neuron to synaptic weights w_{ki} between LIF neurons is achieved in the following fashion. The PSP of a LIF neuron with conductance-based synapses in the HCS has the approximate shape [*Bytschok*, 2011]:

$$V_{\mathrm{PSP}}(t) = \frac{w_{ki}\left(E_k^{\mathrm{rev}} - \overline{V_{\mathrm{eff}}}\right)}{C_{\mathrm{m}} \cdot \left(\frac{1}{\tau_{\mathrm{syn}}} - \frac{1}{\tau_{\mathrm{eff}}}\right)} \cdot \left[\exp\left(-\frac{t - t_{\mathrm{spike}}}{\tau_{\mathrm{eff}}}\right) - \exp\left(-\frac{t - t_{\mathrm{spike}}}{\tau_{\mathrm{syn}}}\right)\right] \,, \tag{2.72}$$

with the synaptic weight w_{ki}, the synaptic time constant τ_{syn} and the mean effective membrane potential from Equation 2.49. For both the LIF domain and the abstract domain, a presynaptic spike must have the same impact on the postsynaptic neuron. This is realized by matching the integrals of individual PSP:

$$\underbrace{\frac{1}{\alpha}\int_0^{\tau_{\mathrm{ref}}} V_{\mathrm{PSP}}(t)\mathrm{d}t}_{\text{LIF neuron}} \stackrel{!}{=} \underbrace{W_{ki} \cdot \tau_{\mathrm{ref}}}_{\text{Stochastic neuron}} \,. \tag{2.73}$$

Evaluating the integral in Equation 2.73 results in the following weight translation between the abstract and the LIF model:

$$W_{ki} = \frac{w_{ki}\left(E_k^{\mathrm{rev}} - \mu\right)}{\alpha C_{\mathrm{m}} \tau_{\mathrm{ref}} \left(\frac{1}{\tau_{\mathrm{syn}}} - \frac{1}{\tau_{\mathrm{eff}}}\right)} \cdot \left[\tau_{\mathrm{syn}}\left(\mathrm{e}^{-\frac{\tau_{\mathrm{ref}}}{\tau_{\mathrm{syn}}}} - 1\right) - \tau_{\mathrm{eff}}\left(\mathrm{e}^{-\frac{\tau_{\mathrm{ref}}}{\tau_{\mathrm{eff}}}} - 1\right)\right] \,. \tag{2.74}$$

Therefore, the weight translation factor β from the abstract domain to the LIF domain

$$w_{ki} = \beta \cdot W_{ki} \tag{2.75}$$

can be expressed as

$$\beta = \frac{\alpha C_{\mathrm{m}} \tau_{\mathrm{ref}} \left(\frac{1}{\tau_{\mathrm{syn}}} - \frac{1}{\tau_{\mathrm{eff}}}\right)}{\left(E_k^{\mathrm{rev}} - \mu\right)} \cdot \left[\tau_{\mathrm{syn}}\left(\mathrm{e}^{-\frac{\tau_{\mathrm{ref}}}{\tau_{\mathrm{syn}}}} - 1\right) - \tau_{\mathrm{eff}}\left(\mathrm{e}^{-\frac{\tau_{\mathrm{ref}}}{\tau_{\mathrm{eff}}}} - 1\right)\right]^{-1} \,. \tag{2.76}$$

By definition, in the abstract neural model from Section 2.1.5, the rectangular PSPs are *renewing*, which means that the contribution to the membrane potential from an afferent neuron is binary (see Equation 2.22). In the context of LIF neurons, however, static synapses lead to additive conductance courses and thus nonlinearly additive superpositions of PSPs.
The TM mechanism (see Section 2.1.6) offers a possible workaround for LIF neurons. Setting $\tau_{\text{facil}} = 0$ in Equation 2.44 leads to $u_{n+1} = U_0$. With the choice $U_0 = 1$ and $\tau_{\text{rec}} \approx \tau_{\text{syn}}$ for example, the parameter R, which describes the recovery of the synaptic strength after emission of an action potential, yields

$$R_{n+1} = 1 - \exp\left(-\frac{\Delta t}{\tau_{\text{syn}}}\right), \tag{2.77}$$

where Δt is the time interval between the n^{th} and the $(n+1)^{\text{th}}$ spike and τ_{syn} the synaptic time constant of the LIF neuron. The condition in Equation 2.77 is equivalent to approximately renewing PSPs.

2.2 Neuromorphic Hardware

With increasing knowledge about the dynamics of biological neural networks it became clear that the operation principles of the brain are fundamentally different from those of traditional von Neumann machines. The question therefore naturally arose to which extent traditional computing architectures are useful instruments for implementing large-scale, biologically inspired models of cortical tissue. As an alternative to the von Neumann architecture, the so-called "neuromorphic" concept was proposed in the early 80s. In its widest sense, as it is used today, the term "neuromorphic" describes a computational device which mimics some aspect(s) of biological neural networks, be it in terms of architecture or dynamics. In a more narrow sense, as originally intended by its original proponents (see e.g. *Mead and Mahowald* [1988]; *Mead* [1989]), neuromorphic devices represent physical implementations of their biological archetype, i.e., they contain electronic components which behave equivalently to neurons and synapses, albeit on a different time or voltage scale.
By adopting design principles directly from biological brains, several important functional consequences are expected. Massively parallel by nature, a physical model of brain tissue should exhibit a similar defect tolerance, much in contrast to traditional computers. Furthermore, the inherent reduction in power consumption would foster excellent scalability, with the potential to emulate large cortical areas, or even an entire brain. Through miniaturization, VLSI technology also offers an additional advantage: due to the small size of the involved components, neuromorphic circuits can operate with a significant speedup compared to the biological circuits they emulate, with obvious benefits for large parameter sweeps or long simulation (or rather, emulation) runs [*Schemmel et al.*, 2010]. Within the FACETS project and its successor BrainScaleS, several such neuromorphic platforms have been developed. The current wafer-scale system can emulate networks with up to 180,000 neurons and 40 million synapses [*Schemmel et al.*, 2010]. However, it

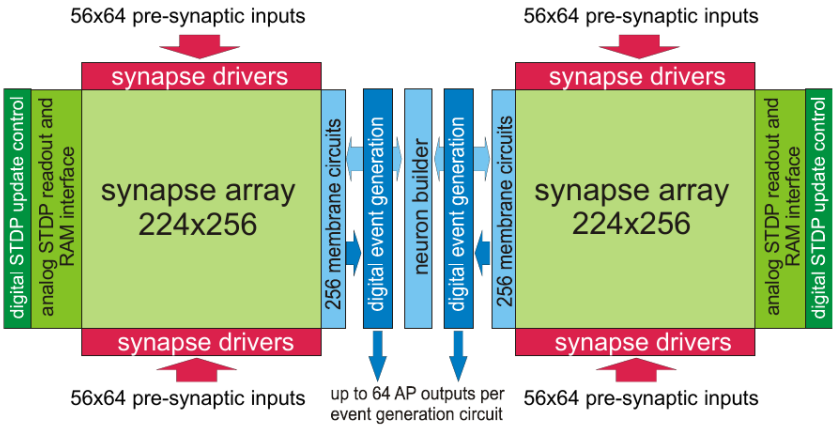


Figure 2.10: Schematic diagram of the Analog Network Core (ANC) from *Schemmel et al.* [2010]. The 512 membrane circuits, the so-called *DenMems*, are located at the center of the chip. The largest part of the chip is covered by two synapse arrays of 256 rows and 224 columns. One half of the membrane circuits receives input from the left synapse array, the other half from the right one. Each membrane circuit receives input via two different synapse lines. Each column of the synapse array contains a synapse driver which can receive up to 64 different presynaptic inputs. The synapse driver converts the digital spike event into an analog current pulse and transmits it to the appropriate membrane circuit. The *neuron builder* can combine multiple DenMems to neurons with multiple compartments. Weight adaptation is implemented in terms of short-term depression and facilitation (STP) and spike-timing dependent plasticity (STDP).

is designed in a scalable fashion, allowing the interconnection of multiple such wafers to form a significantly larger neuromorphic machine.
At the time this thesis is written, the infrastructure of this large-scale neuromorphic system is still in active development. However, a demonstrator setup is available, which consists of the centerpiece of the prospective hardware system, the so-called HICANN chip, and the identical communication infrastructure which will be present in the future system [*Schemmel et al.*, 2012]. This setup has been used for the experiments described in Chapter 6.
In the following, we will start with a detailed description of the circuitry of the HICANN chip in Subsection 2.2.1. Afterwards, in Section 2.2.2, the components of the demonstrator setup are described. Finally, Subsection 2.2.3 gives a brief outlook on the Hybrid Multiscale Facility (HMF), a combination of a large-scale neuromorphic system and a high performance computing part, which is being currently developed.

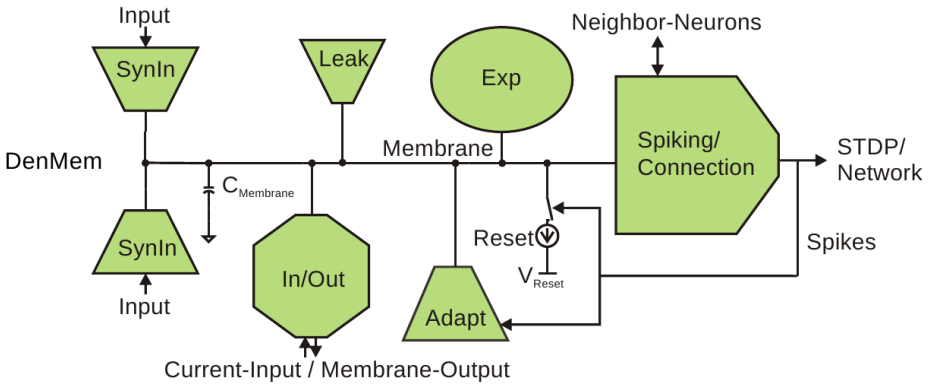


Figure 2.11: The modular structure of a membrane circuit of the ANC. Each module represents a subcircuit which models the respective variables and/or dynamics of the AdEx neuron (see Section 2.1.6). Additionally, dedicated subcircuits allow the injection of an external current or the readout of the membrane potential. Each generated spike is fed back into the circuit to reset the membrane potential and increase the adaptation current. The figure is taken from *Schemmel et al.* [2010].

2.2.1 The HICANN Chip

The HICANN chip is the centerpiece of the neuromorphic hardware system developed within the frame of the *BrainScaleS* project [*Schemmel et al.*, 2010]. It consists of two parts: the Analog Network Core (ANC) and the surrounding digital bus system. The ANC contains the membrane circuits with floating gate cells, which store the parameter values of the model, and two synapse arrays, which manage the transport of analog current pulses to the membrane circuits and occupy the largest part of the ANC. The ANC is surrounded by a bus system, the so-called Layer 1 (L1), which regulates the correct transduction of digital spikes between two neurons on the same HICANN chip as well as between neurons located on adjacent chips.
The following paragraphs offer a detailed description of the ANC, the neuron and synapse integration and the digital circuitry on the HICANN chip. The major information is taken from *Schemmel et al.* [2006], *Schemmel et al.* [2010], *Millner* [2012] and the internal HICANN documentation [*Schemmel et al.*, 2014]. These details are of central importance to the experiments in Chapter 6.

The Analog Network Core of the HICANN chip Figure 2.10 displays a schematic of the ANC. The ANC contains 512 membrane circuits, the so-called *DenMems*, which emulate the dynamics of the AdEx neuron model (see Section 2.1.6). The circuits are divided into two subgroups: one group of 256 circuits receives synaptic input from the left synapse array, the other group from the right synapse array. The 512 circuits on the chip are subdivided into 8 *blocks* of 64 neurons each. The *neuron builder* can combine up to 64 DenMems of the same block to single neurons with multiple compartments.

The largest part of the die is occupied by two synapse arrays, each of which consists of 256 rows and 224 columns. Each column contains one synapse driver which can receive up to 64 different inputs from the same and adjacent HICANN chips. In total, about 115,000 synapse circuits are integrated on one HICANN chip. Each membrane circuit receives analog input via two different synapse lines for which the reversal potentials can be set individually.
The following paragraphs outline the integration of the membrane and synapse circuits and the digital circuitry on the HICANN chip.

Neuron Integration Figure 2.11 illustrates the modular architecture of a membrane circuit of the HICANN chip [*Schemmel et al.*, 2010]. Each DenMem consists of several spatially divided subcircuits, each of which emulates one particular component of the dynamics of the AdEx model (see Section 2.1.6). These are:

- two synaptic input circuits,
- a leakage circuit,
- an exponential circuit, which models the rise of the membrane potential during the upswing of an action potential,
- an adaptation circuit, which models the adaptation current
- and a circuit which generates digital spike events whenever the membrane potential crosses some defined threshold.

The digital spike signal is sent to the digital part of the chip. Additionally, the same spike signal is fed back into the membrane circuit to reset the membrane potential and the adaptation current. An additional subcircuit allows the injection of an external current and the readout of the membrane potential of single neurons via two analog readout channels [*Millner*, 2012]. One of two different membrane capacitances can be selected for one block of membrane circuits [*Schemmel et al.*, 2014].
All 18 parameters of the AdEx model are stored as 24 electrical parameters in 10-bit analog floating gate cells (see Table A.6) which are also located in the center of the ANC and which bias the membrane circuits [*Srowig et al.*, 2007], depending on their voltage (0 to 1.8 mV) or current (0 to 2.5 μA) value. The parameters which are represented by the floating gate cells can be subdivided into *individual* neuron parameters and *global* parameters which are shared by the membrane circuits of one block. The floating gate cells are organized in 4 arrays, each containing 129 columns with 24 rows. Of these, 128 columns are used to store individual neuron parameters, while the last column stores the values for global parameters such as the reset voltage of the block of neurons, the maximum synaptic conductance of the synapse row and the STDP parameters. The main advantage of the floating gate technique is the slow decay of the stored value which is in the range of several hours [*Srowig et al.*, 2007].
However, the current design of the system entails several drawbacks which are crucial for this thesis. One drawback of the complex circuitry are *leakage currents*. Each of

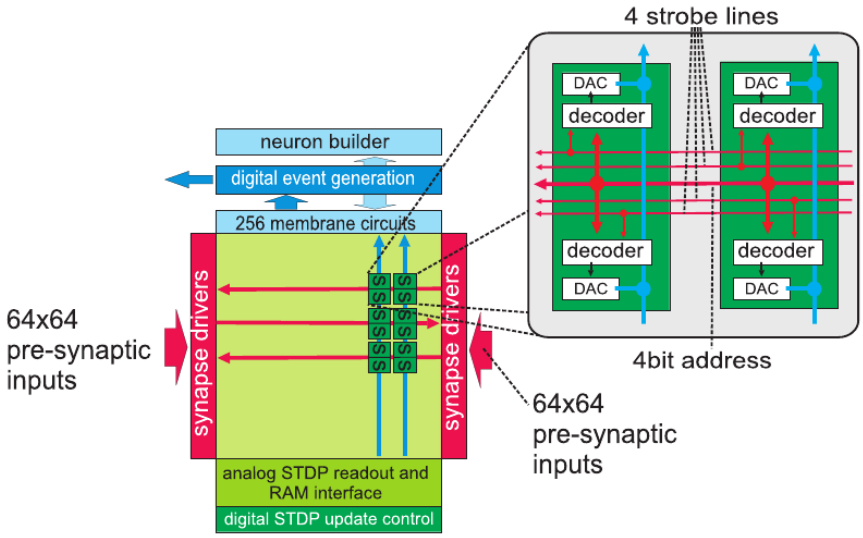


Figure 2.12: Simplified block diagram of a synapse array of the HICANN chip, on the left, and a schematic of the connection between the synapse driver and the synapse lines, on the right. Each synapse driver receives digital spike events, consisting of a 6 bit address, from up to 64 different presynaptic sources. The first 2 bits are used to address one of the four strobe lines, while the last 4 bits determine the destination DenMem of the spike. An activated synapse line sends a square current pulse of an adjustable length τ_{STDF} and amplitude $w_i(t) \times g_{\max}(t)$, where w_i is the individual 4 bit synaptic weight and $g_{\max}$ the row-wise maximum conductance. The figure is taken from *Schemmel et al.* [2008].

the subcircuits generates unwanted leakage currents, which are of significant amplitude compared to functionally relevant currents, such as those generated by synaptic interaction or the LIF leak current $g_{\mathrm{l}}\,(V - E_{\mathrm{l}})$ itself.
In summation, these effects may completely disturb the expected membrane dynamics. Due to varying leakage currents, a complete disconnection of single subcircuits, such as a reduction of the AdEx model to the simpler LIF model by "turning off" the exponential circuit, can not be ensured for every membrane circuit on the HICANN chip. One further drawback lies in the limited precision of the floating gate cell programming, which never reaches the ideal value of 10 bits. Reprogramming the floating gate cells always leads to small deviations in the resulting voltages and currents, thereby causing significant trial-to-trial variability.

Synapse Integration The integration of the synapse circuits is addressed in *Schemmel et al.* [2008] in detail. The following passages summarize the most important aspects.
The output of the membrane circuit is a digital spike event which consists of a time stamp and a 6 bit address. The time stamp is used for the correct recording of the spike events and for Spike-Timing Dependent Plasticity (STDP) [*Song et al.*, 2000; *Friedmann*, 2013],

while the address is used for routing the spike to the correct target neurons. The first 2 bits of the address are used to choose one of four strobe lines, which initiate current pulses of length τ_{STDF}. The duration of these strobe pulses can be modulated by a STP mechanism (see Section 2.1.6). The last 4 bits of the address are transmitted to the synapse array and used to address the 256 DenMem circuits of a half of the HICANN chip. Figure 2.12 depicts a block diagram of a synapse array of the HICANN chip and a schematic of the connection between the synapse driver and the synapse lines.
The synaptic weights $w_i(t)$ are stored by a 4 bit SRAM cell. A 4 bit DAC translates the weight $w_i(t)$ into an output current pulse. The maximum conductance $g_{\max}$, which can be set row-wise by a programmable analog floating-gate cell, determines the scale for the synapse DAC [*Schemmel et al.*, 2008]. The output signal of a synapse is a square current pulse with amplitude $w_i(t) \times g_{\max}(t)$ and duration τ_{STDF}.
The synapses are connected to the DenMem circuits via 2 lines (blue lines in Figure 2.12), respectively. The current pulses of all synaptic connections to one DenMem circuit sum up to two time-varying total input currents. The DenMem circuit translates the total input currents into time-varying conductance traces, emulating two different groups of ion channels, the reversal potentials of which can be programmed arbitrarily [*Schemmel et al.*, 2008]. For instance, one ion channel can represent the excitatory input and one the inhibitory input.
One drawback of the synaptic circuit induced by the nonlinearity of conductance-based models is the difficulty to calibrate the synaptic drivers. The correct calibration of the synaptic drivers requires a successful calibration of the synaptic reversal potentials of the membrane circuits. At the time this thesis is written, the calibration of the synaptic drivers and the reversal potentials is in active development. Synaptic weights have to be set by hand based on the PSPs they generate on the postsynaptic membrane, which obviously leads to limited precision and reproducibility.
Furthermore, a mismatch of the parameter range of the maximum synaptic conductance $g_{\max}$ shrinks the usable input current pulse height [*Millner*, 2012]. Due to this mismatch, in the case of measurements with large biasing parameters in the floating-gate cells, the membrane potential saturates at some maximum and minimum voltage because of saturating synaptic weights. This mismatch will be crucial for the results in Chapter 6 of this thesis.

Digital Circuitry In the following, it is important to differentiate between the hardware time frame (HT) and the equivalent biological time (BT). The ratio between the two is defined by the acceleration factor of the hardware, which is usually assumed as 10^4 [*Schemmel et al.*, 2010]. The membrane circuits generate digital spike signals which are synchronized to an internal reference clock. The frequency of the reference clock is 250 MHz and thus allows a temporal resolution of 4 ns HT (= 40 μs BT). Spikes, which are emitted closer to each other, are shifted in time. These digital signals then have to be routed to the appropriate destination. One possible destination are synaptic drivers on the same or adjacent HICANN chips. For this, the asynchronous serial on-chip Layer 1 (L1) routing is used, which is composed of parallel Low Voltage Differential Signaling (LVDS)

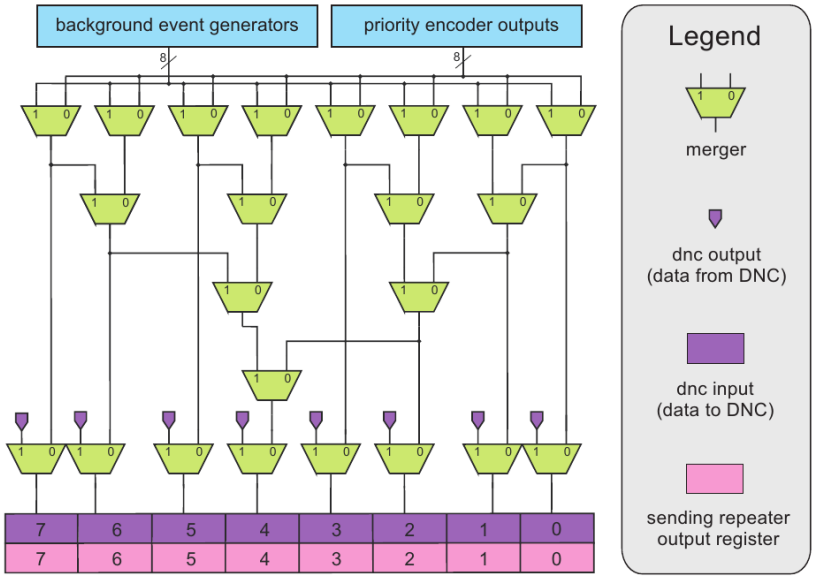


Figure 2.13: The *merger tree* handles the routing of digital spike events on the HICANN chip. The tree structure allows to combine outputs from the membrane circuits and the on-chip background event generators in an arbitrary fashion. The lower row of the merger tree contains the so-called *DNC mergers*. The DNC mergers establish a communication interface with the DNC: Spike events can be sent from the HICANN chip to the DNC to record spike patterns, as well as from the DNC to the HICANN chip to enable the generation of a pre-defined spike input for the membrane circuits. The figure is taken from *Kononov* [2011].

lines. Signal repeaters and switch matrices ensure that the signal reaches its target. The delays of the synaptic connections between two neurons are fixed by design to about $0.12\,\mu$s HT ($= 1.2\,$ms BT).
The other possible destination is the so-called Digital Network Chip (DNC). The DNC is a digital ASIC which allows to record the spike patterns of neurons of the HICANN chip and send them via an FPGA board to the host computer [*Schemmel et al.*, 2010]. Conversely, it allows to generate a predefined spike pattern on the host computer and route it to arbitrary membrane circuits. The DNCs together with the FPGA boards constitute the so-called Layer 2 (L2) routing, which is synchronous and packet-based.
The so-called *merger tree*, which is depicted in Figure 2.13, enables an efficient routing of digital signals on the HICANN chip. There are 8 different mergers in the upper row of the merger tree, one for each block of membrane circuits. These mergers allow to merge the digital spike events of the neurons with spike events from one of 8 Background Event Generators (BEG) which are placed on the HICANN chip.
The BEGs can generate periodic or pseudo-random Poisson-distributed spike patterns

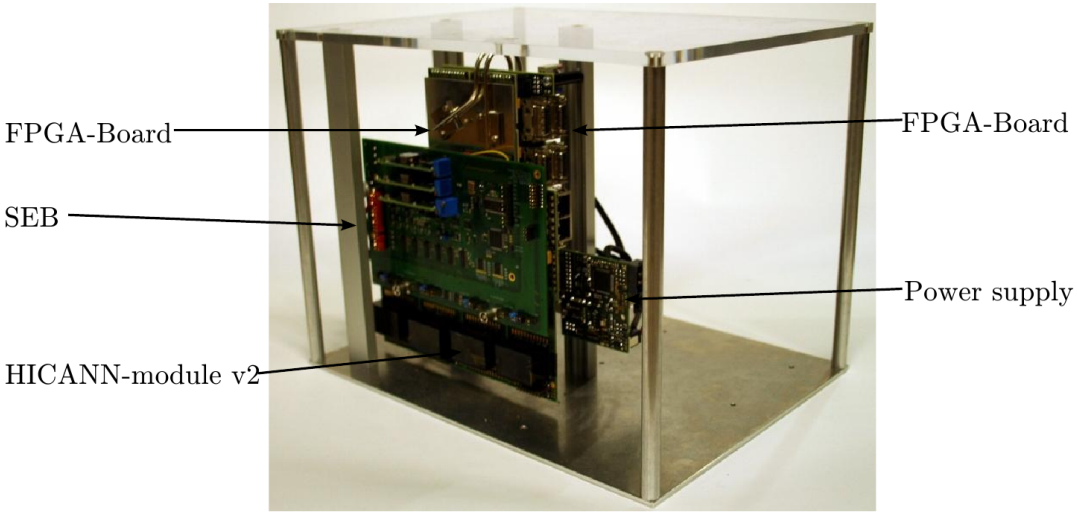


Figure 2.14: The demonstrator setup, which was used to perform the hardware experiments in the course of this thesis [*Schemmel et al.*, 2012]. It consists of a System Emulator Board (SEB) which connects up to 8 HICANN chips to a DNC. Two FPGA boards provide the connection to the host computer and the power supply. Additionally, an ADC board (not shown here) is located on the SEB which allows the simultaneous digital readout of the membrane potential of two neurons of one HICANN chip. The demonstrator setup establishes a communication infrastructure equivalent to that of the FACETS Wafer-Scale Hardware System, the infrastructure of which is currently in active development. The figure is taken from *Millner* [2012].

with a temporal resolution of 4 ns HT [*Kononov*, 2011]. The pseudo-random input is generated via a 16-bit Linear Feedback Shift Register (LFSR). Therefore, the generated Poisson-distributed spike pattern is completely deterministic and repeats after it has occupied its $2^{16} = 65535$ states, which is equivalent to about 262 μs HT (= 2.62 s BT) [*Schemmel et al.*, 2014]. Furthermore, each of the 8 BEGs on one HICANN chip occupies the same sequence of states. However, the starting point of this sequence can be varied by choosing an arbitrary seed between 0 and 65535.
The lower row of the merger tree (see Figure 2.13) contains the so-called *DNC mergers*. At this point, digital signals can leave the HICANN chip towards the DNC or sent to the HICANN chip via the DNC.
The following section presents the hardware setup which was used to perform the experiments in Chapter 6. Particular focus is placed on the achievable data rates between the digital and the analog parts of the system.

2.2.2 Demonstrator Setup

This section offers an overview of the so-called *demonstrator setup* [*Schemmel et al.*, 2012] which was used to execute the experiments in Chapter 6 of this thesis. The demonstrator setup aims at providing the identical communication infrastructure to the HICANN chip (see Section 2.2.1) that will be present in the prospective Hybrid Multiscale Facility (HMF, see Section 2.2.3).
In the following, first, the demonstrator setup will be described. Afterwards, the communication infrastructure between the HICANN chip and the host computer will be outlined with a major focus on the achievable communication bandwidth of the setup.

Setup Figure 2.14 illustrates the demonstrator setup, a scaled-down version of the FACETS Wafer-Scale Hardware System, which is currently in active development [*Schemmel et al.*, 2012]. The setup consists of a System Emulator Board (SEB), 2 FPGA boards, up to 4 DNCs and up to 8 HICANN chips. The HICANN chips are bonded to the SEB, which carries the DNCs. The SEB is plugged into the so-called *iBoard*, which provides the necessary power supply and the clock for the HICANN chip [*Schwartz*, 2012]. The *iBoard* is connected to another FPGA board which handles the communication with the host computer via Ethernet. Furthermore, an ADC board has been added to the setup, which allows the simultaneous digital readout of the membrane voltage of two arbitrary neurons of the HICANN chip.

Communication Infrastructure The communication bandwidths of the prospective HMF system (see Section 2.2.3) and, therefore, also of the demonstrator setup are addressed in *Hartmann et al.* [2010]. The following passages summarize the most important aspects.
The FPGA board which handles the communication of the HICANN chip with the host computer has a 2 GBit/s Ethernet connection to the host, with a communication packet size of 64 Bit, containing 2×28 Bit pulse events and additional configuration bits. Assuming 10 Hz pulse rate in biological real time for each of the 512 possible inputs to a single HICANN, a total rate of 1.6 Gevents/s has to be transmitted, which is an order of magnitude more than can be handled by the host interface [*Hartmann et al.*, 2010]. Therefore, a 4 GB local playback memory which resides on the FPGA board is used to load the spikes before the simulation is started.
The communication bandwidth between the FPGA board and the DNCs amounts to 8 GBit/s with packet sizes of 128 Bit, containing 4×24 Bit pulse events, configuration bits and error detection bits. The data rate between the DNC and one HICANN chip is 2 GBit/s. At a clock frequency of 250 MHz and packet size of 24 Bit, it allows for the transmission of about 27.75 Mevents/s. This means that, at the standard acceleration factor 10^4 compared to biological real-time, the maximum frequency of a regular spike train which can be transmitted to the HICANN chip is about 2775 Hz (biological frequency). Regarding the experiments in Chapter 6, much lower frequencies in the range of 200 Hz per Poisson stimulus will be applied to ensure that, on the one hand, spikes which are closer to each other than the 4 ns temporal resolution are not shifted in time by more

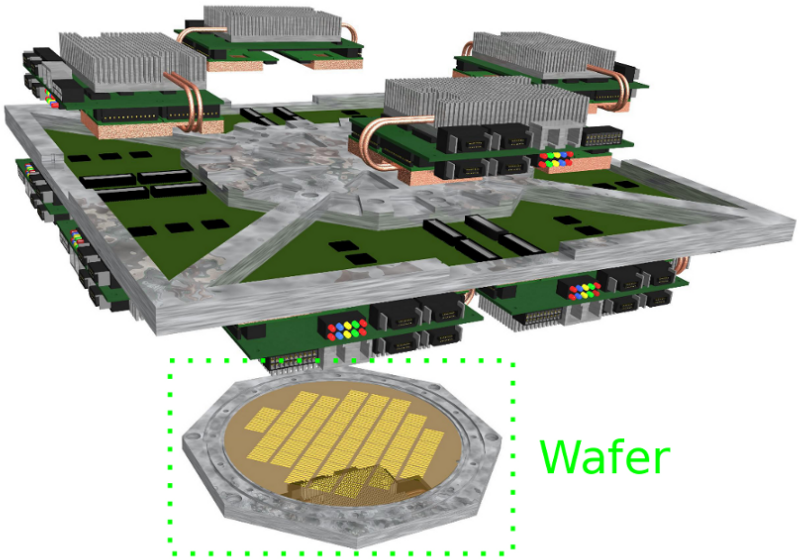


Figure 2.15: The neuromorphic part of the HMF with the wafer as main component. One wafer consists of about 180,000 neurons and more than 40 million synapses. Additionally, the infrastructure of the wafer is depicted which consists of a large PCB with modules which handle the wafer-to-wafer and the wafer-to-host communication. Aluminum frames stabilize the system and act as heat sinks. The figure is taken from [*Brüderle et al.*, 2011].

than about 100 ns hardware execution time. On the other hand, multiple neurons can be simultaneously supplied by multiple independent Poisson sources.

2.2.3 Hybrid Multiscale Facility

The *Electronic Vision(s)* group at the University of Heidelberg and the group for Parallel VLSI Systems and Neural Circuits at the TU Dresden are currently developing the so-called Hybrid Multiscale Facility (HMF), which is a system that goes far beyond the single chip level [*Schwartz*, 2012]. The HMF will consist of two parts: a neuromorphic part of 6 interconnected wafers containing up to 1.2 million membrane circuits and about 260 million synapse circuits, and a modern high-performance computer cluster which provides the communication and control of the neuromorphic system [*Schwartz*, 2012]. The following brief discussion will focus on the characterization of the neuromorphic part of the system, which comprises the wafers and their communication environment.

Wafer The neuromorphic part of the HMF will contain 6 *wafer* modules. Figure 2.15 depicts one of the wafer modules and its associated communication, control and power structures. The silicon wafer has a diameter of 20 cm and is fabricated using the 180 nm

UMC CMOS process [*Schemmel et al.*, 2010]. It consists of 384 HICANN chips (see Section 2.2.1) which are organized in groups of 8 chips, the so-called *reticles*. Each wafer is left uncut after the fabrication and during the so-called *post-processing* step the chips are connected directly on the wafer. With the help of this "third spatial dimension", very large communication densities between the chips are realized. Altogether, one wafer contains about 180,000 membrane circuits and 40 million synapse circuits.

Communication Infrastructure The wafer is connected to a Printed Circuit Board (PCB) which provides the necessary power supply for the wafer and connects it to the digital communication part. The digital communication part has the same hierarchy and identical bandwidths as the communication part of the demonstrator setup described in Section 2.2.2. At the lowest hierarchical level are the 48 DNCs, one DNC per reticle. 4 DNCs are connected to 1 FPGA board, respectively, which amounts to 12 FPGA boards for the whole wafer. The FPGA boards are connected to the host computer and to each other via a 10 GBit/s Ethernet connection. For further information, see *Hartmann et al.* [2010].

2.3 Software Framework

This section describes the software environments which have been used throughout this thesis. It is divided into two subsections.
Subsection 2.3.1 presents the simulator-independent metalanguage PyNN which, together with the simulator NEST [*Diesmann and Gewaltig*, 2002] as its back-end, has been used to run the simulations of networks of LIF neurons in Chapters 3, 4 and 5.
Subsection 2.3.2 deals with the Hardware Abstraction Layer (HAL), the stack of software modules which allows to run experiments on the demonstrator setup described in Section 2.2. In particular, the hardware experiments in Chapter 6 have been executed via the Python-based interpreter PyHAL, which wraps the native HAL language.

2.3.1 Simulation of Neural Networks

Plenty of simulation environments have emerged in computational neuroscience in the past 20 years [*Brette et al.*, 2007] and nowadays serve as non-invasive alternatives for the common *in-vivo* and *in-vitro* experiments in neuroscientific research. All of these software simulators come with their individualities: the level of detail of neuron and network models, the simulation strategy, the algorithms and the native programming language to name some of these. They offer the neuroscientific modeler a diverse pool of tools to set up the desired networks models.
However, with the diversity of software simulators, the lack of reproducibility of neuroscientific experiments increases [*Davison et al.*, 2008]. The code of a model which has been written for one particular simulation environment has to be, in the worst case, completely rewritten in order to fit into another simulator, since network components and parameters are expressed differently in the different environments.

To overcome this lack of reproducibility, the simulator-independent software environment PyNN has been established [*Davison et al.*, 2008] and remains in active development within the *BrainScaleS* project. The following paragraph gives a brief introduction of PyNN and its API. Afterwards, the simulator NEST is described, which is used as PyNN back-end throughout this thesis.

PyNN *PyNN* [2014] is a Python package [*Python*, 2014] for simulator-independent modeling of neural networks. PyNN is designed with the main aim to "write the code once and run it on any supported simulator or hardware device without modification" [*Davison et al.*, 2008].
PyNN exploits the fact that many of the available non-commercial software simulators have, additionally to their native interpreter language, a Python interpreter which offers the equivalent functionality to the native language. Figure 2.16 gives an overview of the simulation and emulation back-ends which PyNN currently supports: NEST [*Diesmann and Gewaltig*, 2002], NEURON [*Hines and Carnevale*, 2003], PCSIM [*Pecevski et al.*, 2009], Brian [*Goodman and Brette*, 2008], NeuroML [*Gleeson et al.*, 2010] and the FACETS and *BrainScaleS* hardware systems, described in Section 2.2. However, due to the still ongoing development of suitable calibration and routing routines of the current neuromorphic hardware system (see Section 2.2), a PyNN-level implementation was infeasible. Instead, the hardware experiments were executed via the native interpreter language StHAL (see Section 2.3.2).
The PyNN API can be divided into two levels of abstraction: the low-level API and the high-level API [*PyNN*, 2014]. The *low-level* procedural API provides a flexible access to individual neurons and synapses via functions like *create()*, *connect()* and *record()*. The *high-level* object-oriented API provides classes like *Population()* and *Projection()*. These classes allow the modeling of neuron populations, layers or columns with arbitrary connectivity algorithms, such as all-to-all connections, random or distance-dependent connectivity.
The PyNN API contains a huge library of models of neurons, synapses and synaptic plasticity which have been proven to show the same dynamics on all supported simulation back-ends [*Davison et al.*, 2008].

NEST The software simulations in the course of this thesis have been carried out using PyNN with the simulator NEST as a back-end.
The NEST software environment [*Diesmann and Gewaltig*, 2002] mainly focuses on the simulation of large networks of neurons with a biologically realistic connectivity pattern, without going into the morphological details of individual neurons. Biologically realistic connectivity patterns require an efficient representation and updating of synapses as well as a parallelization of the network construction and its dynamics. Therefore, the computational efficiency is of major interest for the NEST developers [*Brette et al.*, 2007].

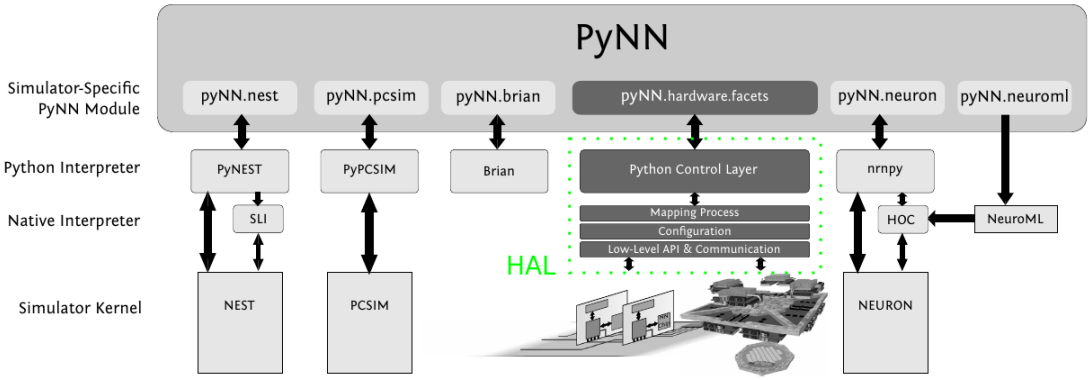


Figure 2.16: Schematic of the simulator-independent modeling language PyNN from *Brüderle et al.* [2011]. The software simulators NEST [*Diesmann and Gewaltig*, 2002], NEURON [*Hines and Carnevale*, 2003], PCSIM [*Pecevski et al.*, 2009], Brian [*Goodman and Brette*, 2008], NeuroML [*Gleeson et al.*, 2010] as well as the FACETS and *BrainScaleS* neuromorphic hardware systems have been integrated into the PyNN concept. The dashed frame comprises the interpreter levels of the *hardware abstraction layer* (HAL), which is currently in active development. The HAL is the stack of software modules which will, in the future, provide an automated translation between PyNN model descriptions and appropriate hardware configuration and control patterns. For further details, see *Brüderle et al.* [2011].

2.3.2 Emulation of Neural Networks

The desired software framework for the neuromorphic hardware setup described in Section 2.2 has to comprise two general aspects: it has to hide as many hardware-specific details as possible from the modeler on the one hand, and, on the other hand, provide an intuitive access to the hardware system similar to that of common software simulators. Therefore, a PyNN interface to the hardware system is currently in active development (see Figure 2.16).
However, setting up an experiment in software distinctly differs from setting up exactly the same experiment on the hardware system. Several additional facts have to be considered when running experiments on hardware: the limited range of neurons and synapses, the maximally available bandwidths, the spatial arrangement of neurons and synapses, varying or even defect dynamics of membrane and synapse circuits due to floating gate imprecisions or production faults during the fabrication process to name just a few.
Therefore, in order to be able to provide a reliable system for the end user, sophisticated mapping and calibration routines have to implemented which are able to translate abstract descriptions of neural networks to maps of membrane circuits on the hardware system and parameters of single neurons, such as reversal potentials and time constants, to their hardware representation.
At the time this thesis was written, the mapping and calibration routines were in active

development. Thus, the experiments which were executed on the hardware system throughout this thesis had to be setup by hand using the native interpreter of the hardware system (HAL, see Figure 2.16). Setting up the experiments by hand involves the searching of appropriate membrane and synapse circuits on the hardware setup and the adjusting of floating gates cells such that the circuits exhibit the expected dynamics. The following paragraph gives a brief introduction to the HAL interpreter and its major classes and functions. Further information can be found e.g. in *Brüderle et al.* [2011].

The HAL Interface The dashed frame in Figure 2.16 comprises the interpreter levels of the so-called *hardware abstraction layer* (HAL). The HAL is the batch of software modules which will provide an automated translation between PyNN model descriptions and correct hardware configuration and control patterns in the prospective hardware system [*Brüderle et al.*, 2011]. In its current version, the hardware system is accessed via calling the native classes and functions of the HAL API from its Python layer, the so-called PyHAL.
At the lowest level, the HAL interfaces with the hardware assembler via various C++ classes and uses the ARQ protocol [*Peterson and Davie*, 2003] to establish a fast and reliable communication. The HAL can be subdivided into two different interfaces: the *stateless HALbe* interface and the *stateful StHAL* interface.
The major task of the HALbe interface is the specification of a coordinate system for each single component of the hardware system described in Section 2.2. The coordinate system represents several addressable layers reaching from wafers, to FPGAs and DNCs connected to the wafer, to reticles on the wafer, to HICANN chips on the reticle, and finally to membrane circuits, floating gate cells, synapse circuits, repeaters and switch matrices on the HICANN chip. For detailed information, see *Schemmel et al.* [2014].
The stateful StHAL interface provides, from the end-user perspective, methods which enable to e.g. write floating gate cells, set up the configuration of the BEGs and the merger tree, change the state of repeaters and switches on the HICANN chip, configure the synapse drivers, inject a current into the membrane of a neuron, read out the membrane potential and the output spikes of a neuron or send spikes from the host computer via the FPGA boards to the chip (see *Schemmel et al.* [2014]).
The Boost.Python library [*Abrahams and Grosse-Kunstleve*, 2003] is used to wrap the original C++ classes and functions to Python, yielding the Python control layer PyHAL [*Brüderle et al.*, 2009]. PyHAL is the highest abstraction level of the hardware system and provides the desired interpreter-based interface to the hardware.
A module for the integration of this interface into the meta-language PyNN (see Section 2.3.1) is implemented, but not utilizable at the time this thesis is written due to the lack of reliable mapping and calibration routines, which are in active development. In the future, the mapping of a PyNN network onto the configuration space of the wafer-scale hardware system will be divided into three automated processing steps: the *placing* of neurons onto the available circuitry, the configuring of the *routing* infrastructure on the device, and the transformation of 18 neuron and synapse parameters into the corresponding 24 hardware parameters [*Brüderle et al.*, 2011].

3 Characterization of LIF Sampling from Boltzmann Distributions

The characterization of LIF sampling from Boltzmann distributions is an essential starting point for the remainder of this thesis. We get used to the crucial parameters of LIF neurons and find out the limits of LIF sampling from Boltzmann distributions. Here, we especially focus on the question: how reliable are the sampling results of LIF neurons from Boltzmann distributions with large weights and biases? Since both implementations of Bayesian Networks with spiking neurons require large weights and biases (see Section 2.1.5) these results will be useful for prognosticating the expected performance of LIF sampling from Bayesian Networks.
Section 3.1 describes the necessary calibration procedure which aims at finding the correspondence of the effective membrane potential of a conductance-based LIF neuron and the probability to find the neuron in the refractory state. For this, the effective membrane voltage of a LIF neuron in a spiking noisy environment will be varied and the time fraction during which the neuron is refractory will be measured. The calibration allows to set the resting potential of a LIF neuron according to the a priori known bias of the associated RV.
Then we attend in Section 3.2 to Boltzmann Machines (BMs) of 5 RVs and test the sampling performance of LIF neurons for randomly generated Boltzmann distributions with different maximal weights and biases. For this, the parameters of the LIF neurons are set with respect to simplicity and compatibility with fluctuations of the neuromorphic hardware which will be used in Chapter 6. The same measure of the sampling quality is performed for stochastic spiking neurons with ideal rectangular or alpha-shaped PSPs (from *Pecevski et al.* [2011]) for reasons of comparison. In contrast to the ideal PSPs, LIF PSPs are highly asymmetric and decay exponentially, thus, they significantly affect the subsequent course of the membrane potential. These facts will turn out to delimitate the LIF sampling performance.
Two randomly picked BMs, one with low weights and biases and one which incorporates very large weights and biases, are then used in Section 3.3 to explore the sampling performance under different parameter choices. The time constants τ_{m}, τ_{syn} and τ_{ref} are in the major focus of this investigation since they define the shape of a PSP. Although it will be impossible to draw general conclusions for all BMs from these results, we will see that there are indeed individual task-dependent parameters, which enable LIF neurons to sample with a satisfactory precision from Boltzmann distributions with large weights and biases.

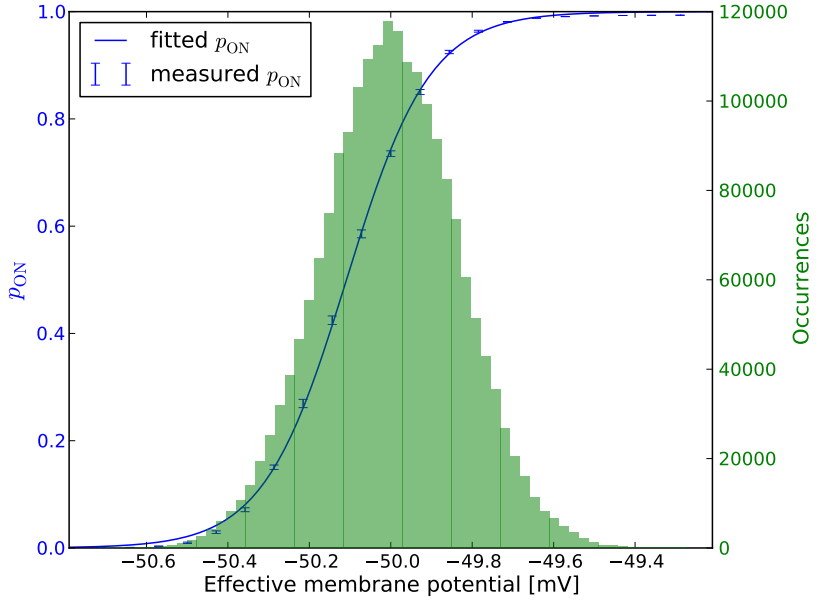


Figure 3.1: The figure shows the calibration procedure which has to be established once per set of neuron parameters. For 21 different effective potentials V_{eff} (*blue*) the fraction of time is measured which the neuron spends in the refractory state (ON-state). The error bars are standard deviations from 5 simulations of duration 200 s. The calibration via effective potentials V_{eff}, instead of resting potentials V_{rest}, is appropriate for the conductance-based model, since it considers the impact of the additional synaptic conductances provided by the background input on the membrane potential. The *green* histogram shows the distribution of the free membrane potential of a LIF neuron with parameters from Table A.1 and $V_{\mathrm{rest}} = -50\,\mathrm{mV}$.

3.1 Calibration of a LIF Neuron to Perform LIF Sampling

A conductance-based LIF neuron has to be calibrated before it can be used for sampling from probability distributions. Similar to the NCC in Equation 2.18, its mean membrane potential has to be related to its probability p_{ON} to be in the refractory state.
For this, two essential steps have to be fulfilled. First we need to measure the activation curve. A range of effective membrane potentials V_{eff} is chosen which covers the whole range of probabilities p_{ON} from 0 to 1. This is done by estimating the mean μ_V and the spread σ_V of the free membrane potential based on the known mean synaptic input. In the second step, these effective membrane potentials are translated into resting potentials V_{rest} and corresponding leakage conductances g_{l}. For each resting potential and leakage conductance, p_{ON} can be measured from a single neuron simulation. A logistic function is then fitted to the measured data and we directly obtain the value of the resting potential

$\overline{V}^{b=0}$, for which $p_{\mathrm{ON}} = 0.5$, and the parameter α from Equation 2.70. This will allow for translating Boltzmann biases and weights to resting potentials of neurons and weights of the synaptic connections.

3.1.1 Finding a Good Range of Effective Membrane Potentials

Section 2.1.7 demonstrates that under balanced random synaptic stimulation the free membrane potential of a LIF neuron can be described by an OU process. The histogram in Figure 3.1 shows that the free membrane potential of the neuron then is Gaussian distributed. Given the parameters of the neuron and the background input, the mean μ_V and the spread σ_V of the free membrane potential amount to (see Equations 2.49 and 2.50):

$$\mu_V = \frac{g_{\mathrm{l}} E_{\mathrm{l}} + \sum_i \overline{g_i(t)} E_i^{\mathrm{rev}}}{\overline{g_{\mathrm{tot}}(t)}} \,, \tag{3.1}$$

and

$$\sigma_V^2 = \sum_i \nu_i S_i^2 \left(\frac{\tau_{\mathrm{eff}}}{2} + \frac{\tau_{\mathrm{syn},i}}{2} - 2\frac{\tau_{\mathrm{eff}}\tau_{\mathrm{syn},i}}{\tau_{\mathrm{eff}} + \tau_{\mathrm{syn},i}} \right) \,, \tag{3.2}$$

with

$$S_i = \frac{w_i \left(E_i^{\mathrm{rev}} - \mu_V\right)}{C_{\mathrm{m}} \left(\frac{1}{\tau_{\mathrm{eff}}} - \frac{1}{\tau_{\mathrm{syn},i}} \right)} \,. \tag{3.3}$$

Now we can estimate the range of effective membrane voltages which is needed to cover the whole range of probabilities p_{ON} from 0 to 1. Under the assumption that in the HCS the membrane potential will stay close to the threshold compared to the reversal potentials the threshold voltage is used as central point for this range. Heuristically $4\sigma_V$ turns out to be a good estimate for the spread of this range: it contains more than 99.9% of the possible assignments of the free membrane voltage and likewise provides for an optimal resolution of the growth of the activation curve. We choose N_V equally spaced potentials with a maximal effective membrane potential

$$V_{\mathrm{eff,max}} = V_{\mathrm{th}} + 4\,\sigma_V \left(1 + \frac{1}{N_V - 1}\right) \,, \tag{3.4}$$

and a minimal effective membrane potential

$$V_{\mathrm{eff,min}} = V_{\mathrm{th}} - 4\,\sigma_V \left(1 + \frac{1}{N_V - 1}\right) \,. \tag{3.5}$$

3.1.2 Measuring the Activation Curve

Now the membrane potential of the LIF neuron has to be set to these effective membrane potentials and the fraction of time has to be measured during which the neuron stays refractory. For this, the effective membrane potentials have to be transformed to resting

potentials V_{rest} and additional leakage conductances w. The maximal and minimal additional conductance $w_{\max}$ and $w_{\min}$ can be calculated:

$$w_{\max} = \frac{\overline{g_{\text{tot}}(t)}V_{\text{eff,max}} - g_{\text{l}}V_{\text{th}} - \sum_i \overline{g_i(t)}E_i^{\text{rev}}}{E_{\text{exc}}^{\text{rev}} - V_{\text{eff,max}}} , \tag{3.6}$$

and $w_{\min} = -w_{\max}$. For each of the equally spaced N_V additional conductances w the total leakage conductance amounts to $\tilde{g}_{\text{l}} = g_{\text{l}} + |w|$. The parameters of the LIF neuron which determine the effective membrane potential are τ_{m} and V_{rest}, for which we get:

$$\tau_{\text{m}} = \frac{C_{\text{m}}}{\tilde{g}_{\text{l}}} , \tag{3.7}$$

and

$$V_{\text{rest}} = \begin{cases} \frac{g_{\text{l}}V_{\text{th}} + wE_{\text{exc}}^{\text{rev}}}{\tilde{g}_{\text{l}}}, & \text{if } w \geq 0 \\ \frac{g_{\text{l}}V_{\text{th}} - wE_{\text{inh}}^{\text{rev}}}{\tilde{g}_{\text{l}}}, & \text{if } w < 0 \end{cases} . \tag{3.8}$$

Following this procedure, we get N_V different values for τ_{m} and V_{rest}. These values can now be initialized for the LIF neuron in simulations. If the neuron elicits N_S spikes in a simulation of duration T, then the probability to be in the ON-state $z = 1$ is

$$p_{\text{ON}} = \frac{N_S\tau_{\text{ref}}}{T} . \tag{3.9}$$

The blue dots in Figure 3.1 illustrate the measured probabilities p_{ON} for $N_V = 21$ different effective membrane potentials for a LIF neuron with parameters from Table A.1. The error bars indicate the standard deviations from 5 simulations of duration 200 s.
The measured data is fitted to a logistic function

$$\sigma\left(\frac{V - \overline{V}^{b=0}}{\alpha}\right) = \frac{1}{1 + \exp\left(-\left(\frac{V-\overline{V}^{b=0}}{\alpha}\right)\right)} , \tag{3.10}$$

and the parameters $\overline{V}^{b=0}$ and α can be directly extracted. Figure 3.1 shows a logistic fit to measured data. For the intermediate part of the curve, the logistic function well describes the data. However, at low voltages the transition to $p_{\text{ON}} = 0$ occurs faster. At large voltages, the state $p_{\text{ON}} = 1$ is never reached because of non-zero time constants.

3.2 LIF Sampling from Boltzmann Distributions of 5 Random Variables

From previous experiments by e.g. *Petrovici et al.* [2013], it is known that the LIF parameters in Table A.1 provide for a good sampling performance from BMs with low weights and biases. We will conduct a closer investigation of BMs via the following experiment: Different BMs of 5 RVs are randomly generated by drawing the weights and biases from a uniform distribution $[-w_{\max}, w_{\max}]$. The parameter $w_{\max}$ is varied

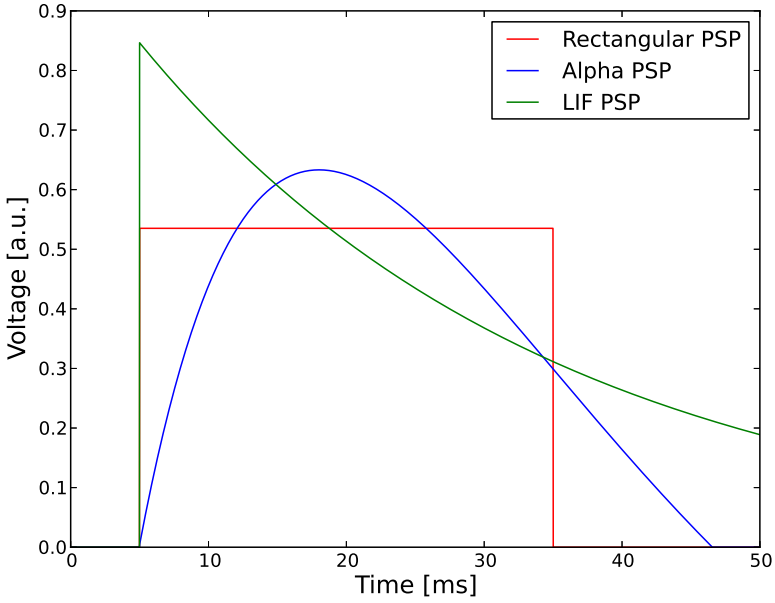


Figure 3.2: The LIF PSP shape (*green*) with $\tau_{\text{ref}} = 30\,\text{ms}$ and $\tau_{\text{syn}} = 30\,\text{ms}$ which was used in the simulations in this section. In addition, the rectangular PSP and alpha-shaped PSP, which are the original PSPs from *Pecevski et al.* [2011], are plotted.

from 0.2 to 10.0 and for each generated BM a simulation of 200 s is run. The obtained joint probability distribution is compared to the theoretical joint distribution via the KL divergence (see Equation 2.15). This experiment is repeated 10 times for each w_{max}.
Figure 3.3 shows the results of the simulations. Furthermore, it shows results of the same experiments with stochastic neurons with ideal rectangular PSPs and alpha-shaped PSPs (see Figure 3.2). For each w_{max}, the LIF sampling result is does not reach the quality of sampling with stochastic neurons: about one order of magnitude less precise than the simulations using alpha-shaped PSPs and up to two orders of magnitude less precise than the results with ideal rectangular PSPs. Furthermore, the sampling quality decreases by three orders of magnitude for large weights and biases, which is to some degree also observable in the experiments with stochastic neurons: The KL divergence between the simulated and theoretical joint probability distribution for alpha-shaped PSPs decreases by two orders of magnitude, for rectangular PSPs by about one order of magnitude.
The imprecise LIF sampling performance can be explained based on the PSP shapes in Figure 3.2. Two differences are crucial: On the one hand, in opposite to the PSPs of the stochastic neurons the LIF PSP is highly asymmetric, thus the firing probability directly after an incoming spike is by far larger than the firing probability at the end of the refractory period. On the other hand, the LIF PSP has an infinite duration and thus influences the course of the membrane potential beyond τ_{ref}. This influence obviously

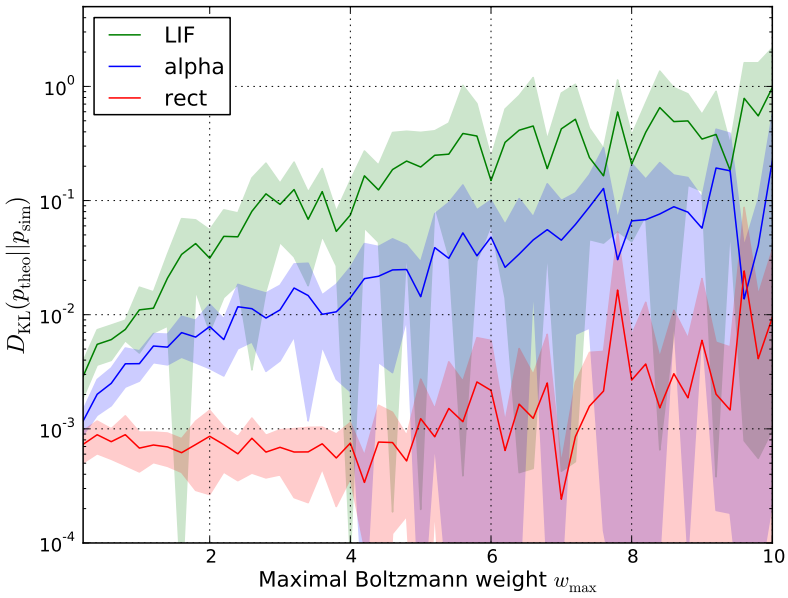
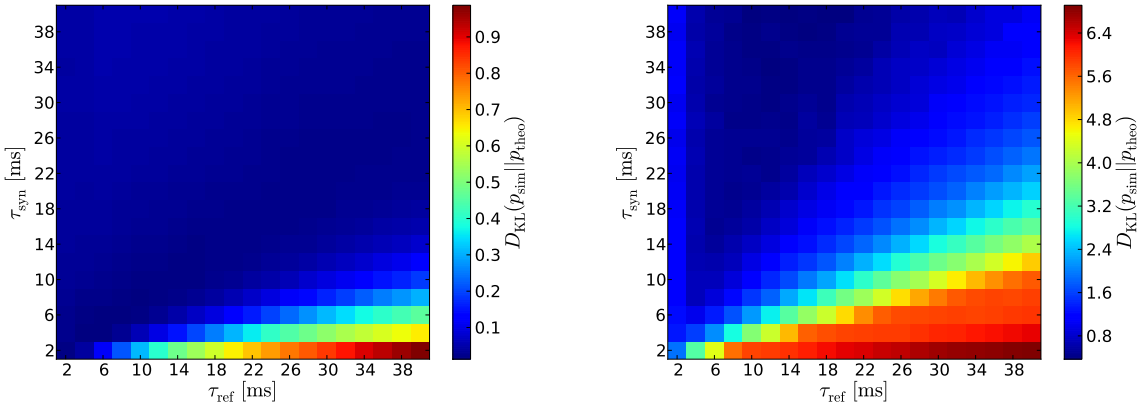


Figure 3.3: Performance measure of simulated Boltzmann machines (BMs) of 5 RVs with neuron parameters from Table A.1. The performance is measured via the KL divergence from Equation 2.15 for the joint probability distributions of the five random variables. The three PSP shapes from Figure 3.2 are used: the LIF PSP for LIF sampling and the rectangular PSP and alpha-shaped PSP for neural sampling with stochastic neurons. Each measured data point results from 10 simulation runs of 200 s each. The confidence areas denote the standard deviations. For each new data point, weights and biases are randomly chosen from the interval $[-w_{\mathrm{max}}, w_{\mathrm{max}}]$.

increases with larger weights.
However, the LIF-based Bayesian networks in Chapter 4 and 5 will require a reliable performance for large weights and biases. Therefore, the following section deals with the exploration of the parameter space of LIF neurons and with the question whether the parameters in Table A.1 are the best parameters for *all* BMs.

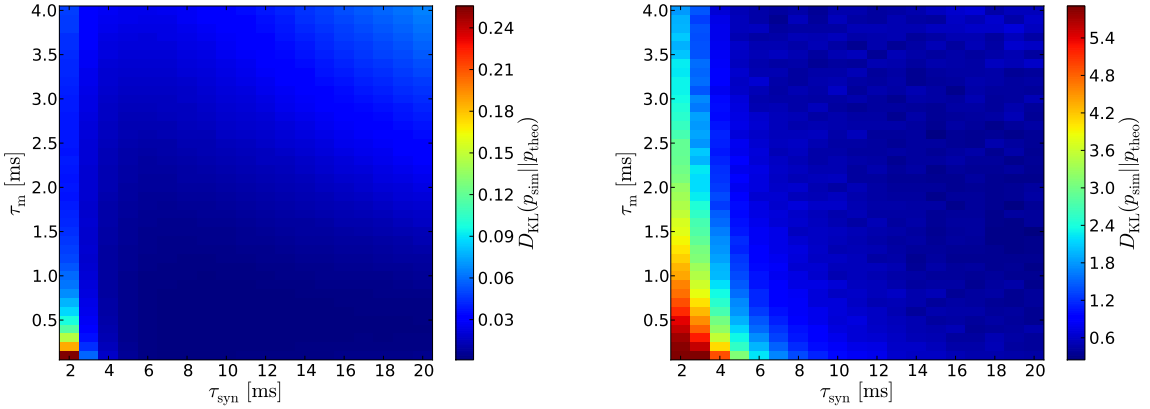
3.3 Parameter Analysis of LIF Neurons for Boltzmann Machines of 5 Random Variables

After some first unsatisfactory sampling results from BMs with large weights and biases in Section 3.2, it is important to gain a sense for the dependence of the sampling performance on the parameters of the LIF neurons. In the following, the standard parameters for LIF neurons which will be used for sampling from BMs are purposely fixed. Afterwards some



A: $w_{\max} = 1.5$ **B:** $w_{\max} = 10.0$

Figure 3.4: Color-coded KL divergence of the simulated and theoretical joint probability distribution for a simulated BM consisting of 5 RVs. The varied quantities are the time constants τ_{ref} and τ_{syn}. Each square corresponds to one simulation of duration $1000\,\mathrm{s} \cdot \tau_{\mathrm{ref}}/30\,\mathrm{ms}$. The BM weights are chosen from the interval $[-w_{\max}, w_{\max}]$ and the BM biases from the interval $[-0.5 \cdot w_{\max}, 0.5 \cdot w_{\max}]$ once, then both are kept fixed during the sweep.



A: $w_{\max} = 1.5$ **B:** $w_{\max} = 10.0$

Figure 3.5: Color-coded KL divergence of the simulated and theoretical joint probability distribution for a simulated BM consisting of 5 RVs with the varied quantities τ_{m} and τ_{syn}. The parameter τ_{ref} is 10 ms. Each square corresponds to one simulation of duration 333 s. The BM weights and biases are the same as in Figure 3.4.

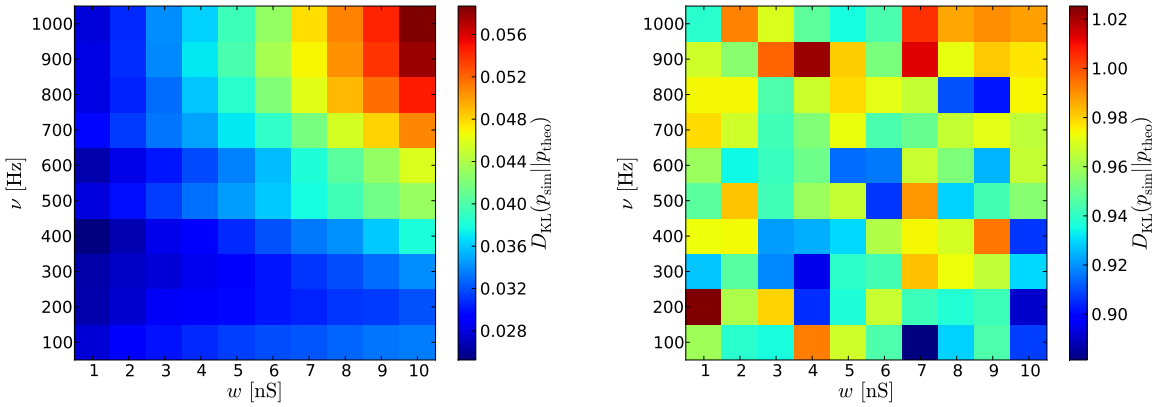
free parameters will be altered and the sampling performance will be measured for the new parameter choices. In particular, it will be tested whether the sampling results can be improved for BMs with large weights and biases by optimizing the neural parameters.

3.3.1 Parameter Selection

The parameters in Table A.1 have been selected for different reasons. First, there is the reason of simplicity. The values for V_{th}, $E_{\text{exc}}^{\text{rev}}$ and $E_{\text{inh}}^{\text{rev}}$ are selected such that EPSPs and IPSPs are approximately the same size. The STP parameters U_0 and τ_{rec} are chosen in order to provide for renewal PSPs (see Section 2.1.7).
Second, for mathematical reasons it is important to keep the quotient $\tau_{\text{eff}}/\tau_{\text{syn}}$ small such that the first-order approximation of Equation 2.69 can be applied. This is achieved by a very small membrane time constant τ_{m} and a very large synaptic time constant τ_{syn}.
Finally, the parameters C_{m}, V_{reset} and the background input rates are chosen such that they match the constraints of the neuromorphic hardware which will be used in Chapter 6.

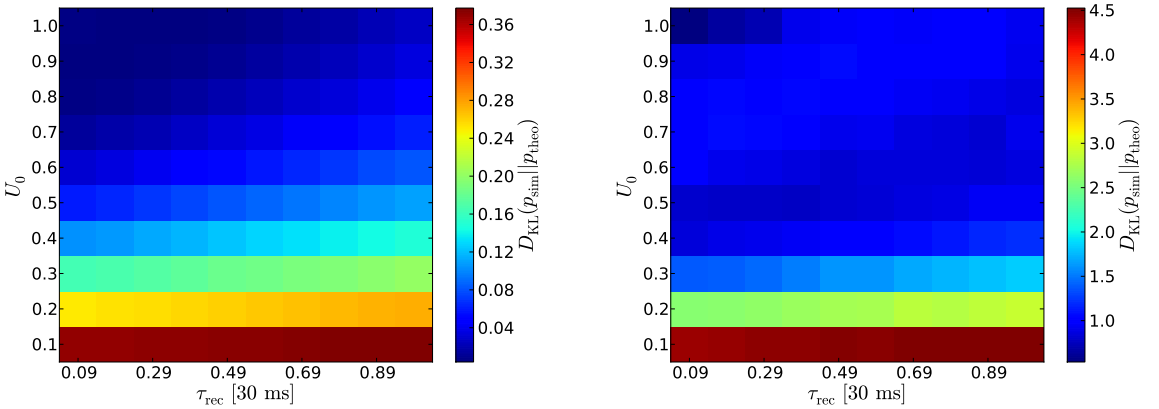
3.3.2 Parameter Optimization for Sample Boltzmann Machines

In the following an extensive parameter analysis of LIF neurons is provided based on the LIF sampling results from two different Boltzmann distributions. The distributions are randomly generated once, by uniformly drawing weights from the interval $[-w_{\max}, w_{\max}]$ and biases from $[-0.5 \cdot w_{\max}, 0.5 \cdot w_{\max}]$, and kept for all experiments. For the first BM the maximum weight is $w_{\max} = 1.5$, so only small weights and biases are allowed.
The KL divergence of the product of theoretical marginals and the theoretical joint distribution is $D_{\text{KL}}(p_{\text{POM}}||p_{\text{theo}}) = 0.02$. This result is taken as a worst case scenario because it assumes independence among the RVs. The second BM has $w_{\max} = 10.0$ and also allows for large weights and biases. Its KL divergence of the product of theoretical marginals and the theoretical joint distribution is $D_{\text{KL}}(p_{\text{POM}}||p_{\text{theo}}) = 2.03$.
If not explicitly mentioned, the standard parameters of the following simulations are taken from Table A.1. The experimental durations of the simulations are chosen for reasons of comparability. The duration of a simulation with LIF neurons with a refractory time $\tau_{\text{ref}} = 30\,\text{ms}$ is fixed to 1000 s for this chapter. For other τ_{ref} the duration of the simulation obviously has to be $1000\,\text{s} \cdot \tau_{\text{ref}}/30\,\text{ms}$ such that the LIF neuron has the same number of opportunities to change the state of its corresponding RV.
Figure 3.4 shows the LIF sampling results for different choices of the refractory time constant τ_{ref} and the synaptic time constant τ_{syn}. In both cases, $\tau_{\text{syn}} \ll \tau_{\text{ref}}$ is the yields a bad sampling quality and $\tau_{\text{syn}} \approx \tau_{\text{ref}}$ performs better. For the BM with $w_{\max} = 10$, $\tau_{\text{syn}} > \tau_{\text{ref}}$ yields the best results of about $D_{\text{KL}}(p_{\text{sim}}||p_{\text{theo}}) = 0.4$ which is distinctly better than $D_{\text{KL}}(p_{\text{POM}}||p_{\text{theo}}) = 2.03$.
Figure 3.5 illustrates the LIF sampling quality under variation of the membrane time constant τ_{m} and the synaptic time constant τ_{syn} with $\tau_{\text{ref}} = 10\,\text{ms}$. For the BM with $w_{\max} = 1.5$, the best results are achieved for $\tau_{\text{m}} = 0.1\,\text{ms}$ and $\tau_{\text{syn}} < \tau_{\text{ref}}$. LIF sampling from the BM with $w_{\max} = 10$, if $\tau_{\text{syn}} > \tau_{\text{ref}}$, a result which can also be seen in Figure 3.4.



A: $w_{\max} = 1.5$ **B:** $w_{\max} = 10.0$

Figure 3.6: The impact of excitatory and inhibitory background rates ν_i and weights w_i on the sampling performance for a BM consisting of 5 RVs. The BM weights and biases are the same as in Figure 3.4. Each square corresponds to a 1000 s-simulation with $\tau_{\mathrm{ref}} = 30$ ms.



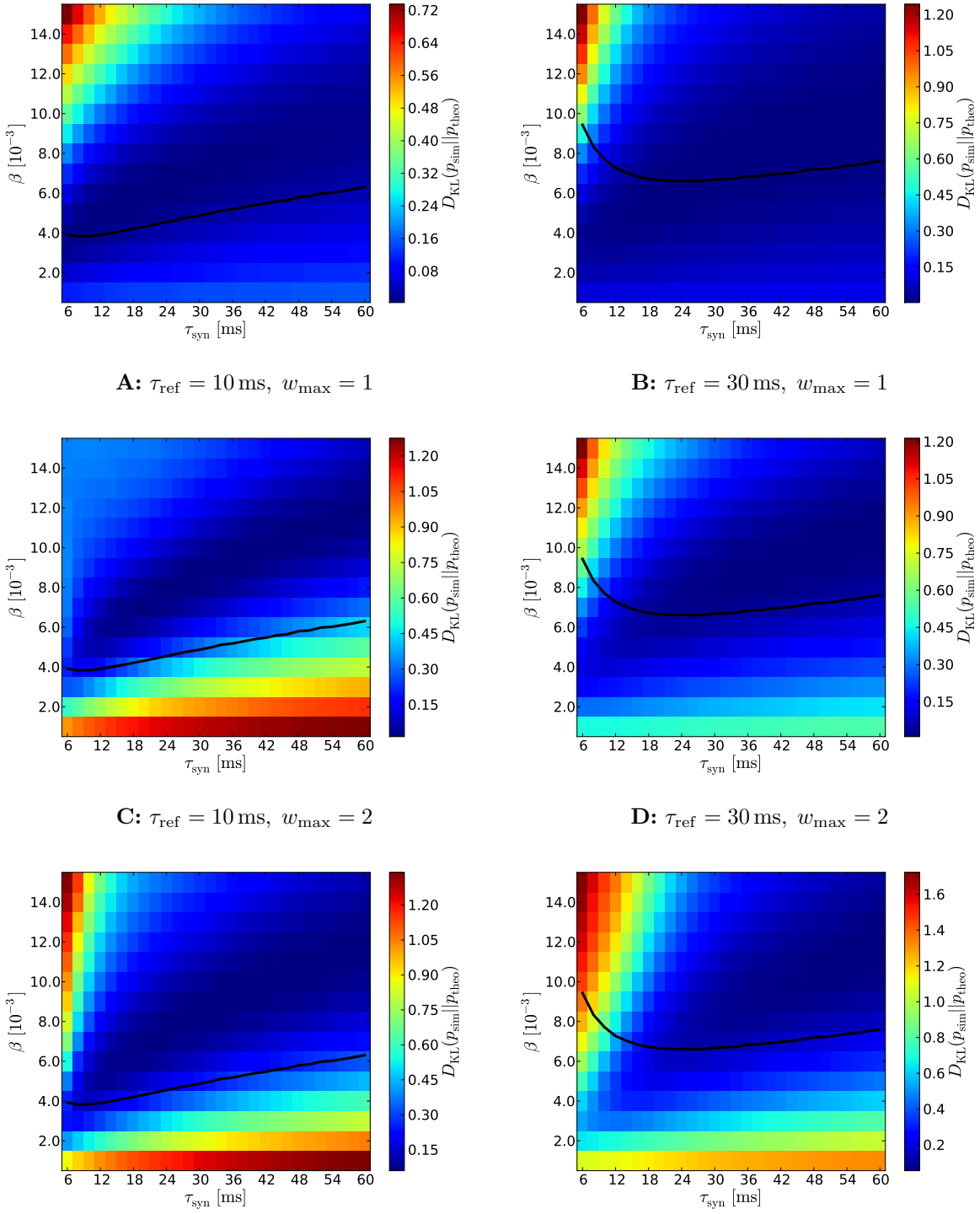
A: $w_{\max} = 1.5$ **B:** $w_{\max} = 10.0$

Figure 3.7: The impact of the STP parameters utilization U_0 and the recovery time constant τ_{rec} on the sampling quality for a BM consisting of 5 RVs. The BM weights and biases are the same as in Figure 3.4. Each square corresponds to a 1000 s-simulation with $\tau_{\mathrm{ref}} = 30$ ms.

Figure 3.6 shows the impact of the input rate and weight of the Poisson background stimulus on the LIF sampling performance. For the BM with $w_{\max} = 1.5$, a tendency towards small weights is visible, an observation which is also expected due to Equation 2.55. For the BM with large weights and biases, no significant preference of a rate and weight can be observed. However, in difference to the parameter sweeps in Figures 3.4 and 3.5, the overall variations of the sampling performance are very low.
Figure 3.7 demonstrates the LIF sampling performance for different choices of the utilization of the synaptic efficacy parameter U_0 and the recovery time constant τ_{rec} from the TM model (see Section 2.1.6). For both BMs, the best results are achieved with the utilization parameter $U = 1$ and the recovery time $\tau_{\text{rec}} = 0.09 \cdot \tau_{\text{syn}}$. These results might express that for these two special BMs the resulting LIF PSPs are too large. In this case, the experimental results would probably improve by choosing a larger weight translation factor β from Equation 2.76.
Figure 3.8 shows the impact of the weight translation factor β from Equation 2.76 and the synaptic time constant τ_{syn} on the LIF sampling quality. Here, three different BMs are examined: a BM with $w_{\max} = 1.0$ in Panels A and B, a BM with $w_{\max} = 2.0$ in Panels C and D and another BM with $w_{\max} = 4.0$ in Panels E and F. The biases are drawn uniformly from the interval $[-0.1 \cdot w_{\max}, 0.1 \cdot w_{\max}]$. The experiments are performed with two different refractory time constants: once with $\tau_{\text{ref}} = 10\,\text{ms}$ and once with $\tau_{\text{ref}} = 30\,\text{ms}$. The color-plots imply that the choice $\tau_{\text{ref}} = \tau_{\text{syn}}$ yields the best sampling results for all three BMs, but only if the weight translation factor β is calculated by matching the areas under the PSP shape as in Equation 2.76 (black curve). However, these are not the best sampling results. There are distribution-specific parameters for which the sampling quality is even better. These are synaptic time constants $\tau_{\text{syn}} > \tau_{\text{ref}}$ together with $\beta \approx 0.01$.

3.4 Conclusion

This chapter showed that there are no unique neuron parameters which yield optimal results for LIF sampling from arbitrary Boltzmann distributions. This circumstance has advantages and disadvantages with regard to the requirements in the next chapter, where we move to LIF sampling from distributions represented as Bayesian Networks.
On the one hand, the results from this chapter do not preclude from a satisfactory sampling quality from BMs with large weights and biases when using LIF neurons. Indeed, there exist individual task-dependent parameters which enable LIF neurons to approximately generate samples from Boltzmann distributions with large weights and biases.
On the other hand, however, for each new Boltzmann distribution an extensive search for the parameters which facilitate the optimal LIF sampling quality has to be accomplished. These parameters can not be calculated analytically and can only be optimized via an exhaustive exploration of the parameter space.



A: $\tau_{\text{ref}} = 10\,\text{ms},\ w_{\text{max}} = 1$

B: $\tau_{\text{ref}} = 30\,\text{ms},\ w_{\text{max}} = 1$

C: $\tau_{\text{ref}} = 10\,\text{ms},\ w_{\text{max}} = 2$

D: $\tau_{\text{ref}} = 30\,\text{ms},\ w_{\text{max}} = 2$

E: $\tau_{\text{ref}} = 10\,\text{ms},\ w_{\text{max}} = 4$

F: $\tau_{\text{ref}} = 30\,\text{ms},\ w_{\text{max}} = 4$

Figure 3.8: Color-coded KL divergence of the simulated and theoretical joint probability distribution for BMs of 5 samplers. Each square corresponds to a 333 s simulation for $\tau_{\text{ref}} = 10\,\text{ms}$ (*left column*) or a 1000 s simulation for $\tau_{\text{ref}} = 30\,\text{ms}$ (*right column*). In each simulation, the synaptic time constant τ_{syn} and the weight translation factor β (see Equation 2.74) are varied. Before the sweep, the BM weights are chosen from the interval $[-w_{\text{max}}, w_{\text{max}}]$ while the BM biases are chosen from the interval $[-0.1 \cdot w_{\text{max}}, 0.1 \cdot w_{\text{max}}]$ and then are kept fixed for the simulations with both refractory times. The *black* lines in the color-plots indicate the calculated β for the given τ_{syn} and τ_{ref}.

4 Bayesian Networks: Implementation 1 with LIF Neurons

The previous chapter focused on the feasibility of LIF sampling from Boltzmann distributions which have large weights and biases. Simulation results showed that the sampling performance varies greatly depending on the choice of the neuron parameters. Now we will focus on LIF sampling from Bayesian Networks (BNs). Section 2.1.5 presented two methods to implement BNs with LIF neurons.
This chapter deals with Implementation 1, which exploits the fact, that each BN can be reduced to a Boltzmann distribution with the help of auxiliary variables. Hereby, each n^{th}-order factor of the probability distribution, with $n > 2$, is represented via 2^n additional auxiliary RVs, each of which codes for one of the possible states of the n^{th}-order factor.
Section 4.1 will explicitly describe the representation via LIF neurons for two example BNs: a model of the Visual Perception Experiment (VPE) and the ASIA network (see Section 2.1.3). The VPE is represented by a probability distribution of 4 RVs and incorporates an explaining away effect. The ASIA network consists of 7 RVs which are arranged in a structure of three layers. This network has two explaining away effects, multiple acyclic loops and a larger discrepancy of prior and conditional probabilities than the model of the VPE.
Section 4.2 presents the simulation results of a rigorous parameter analysis for both example BNs. We will observe that it is indeed possible to find a set of parameters which yields good LIF sampling results for both probability distributions.
In Section 4.3, the optimal LIF sampling results for the two example distributions are illustrated and compared to sampling results of stochastic neurons with rectangular and alpha-shaped PSPs. Although the results for the VPE are comparable to those of the stochastic neurons, sampling from the ASIA network will yield unsatisfactory results for typical inference tasks.
With the special aim to enhance the LIF sampling results for the ASIA network, Section 4.4 introduces a method which enables to create LIF PSPs close to the ideal rectangular shape. Hereby the postsynaptic effect of each sampling neuron is modified by a feed forward chain of additional neurons which are connected on the postsynaptic site of the sampling neuron. It is demonstrated for the two example BNs that these so-called mLIF PSPs yield better sampling results than the standard LIF PSPs. This section also reveals the overall slow convergence of Implementation 1 due to the introduction of additional RVs.
Section 4.5 generalizes the sampling results for arbitrary BNs of 5 RVs. For this, BNs are randomly generated and the extremeness of the discrepancy of the prior and conditional

probabilities is determined by a parameter α. The simulations show that mLIF PSPs outperform the original LIF PSPs for arbitrary BNs.

4.1 Implementation 1: Illustrative Implementation of Example Bayesian Networks

This section introduces the neural implementation of the two Bayesian networks, the VPE and the ASIA network, which will be applied in the following sections as example probability distributions to test the LIF sampling performance. The phenomenology of both Bayesian networks is described in Section 2.1.3.

4.1.1 Implementation of the Visual Perception Experiment

The VPE is modeled by the joint probability distribution

$$p(z_1, z_2, z_3, z_4) = p(z_1)\, p(z_2)\, p(z_3|z_1, z_2)\, p(z_4|z_2)\ . \tag{4.1}$$

Due to the presence of the third-order factor $p(z_3|z_1, z_2)$, 8 auxiliary RVs have to be introduced, which code for the 8 possible assignments of $p(z_3|z_1, z_2)$. Figure 4.1 illustrates the representation of the BN with neurons. In addition to the 4 principal neurons ν_1 to ν_4 which code for the 4 principal RVs z_1 to z_4 there are 8 auxiliary neurons. The connectivity pattern of the neural network is described in Section 2.1.5. Due to Equation 2.27 and the maximal conditional probability of the third-order factor $\max\left(p(z_3|z_1, z_2)\right) = 0.85$, the strength of the connections between the principal and the auxiliary neurons amounts to $M_{\text{exc}} = 8.5$ and $M_{\text{inh}} = -8.5$. The biases of the auxiliary neurons are set according to Equation 2.28. The biases of the principal neurons are calculated via Equation 2.23 and the weights between the neurons ν_2 and ν_4 according to Equation 2.24.

4.1.2 Implementation of the ASIA Network

The ASIA network is modeled by the joint probability distribution

$$p(A, S, T, C, B, X, D) = p(A)\, p(S)\, p(T|A)\, p(C|S)\, p(B|S)\, p(X|T, C)\, p(D|T, C, B)\ . \tag{4.2}$$

The probability distribution has two factors which depend on more than 2 RVs. The factor $p(X|T, C)$ provides for 8 auxiliary RVs, the factor $p(D|T, C, B)$ for 16 auxiliary RVs. The entire neural network which represents this BN is illustrated in Figure 4.2. It consists of 7 principal neurons and 24 auxiliary neurons. With $\max\left(p(X|T, C)\right) = 0.98$ we have the connection strengths $M_{\text{exc}} = 9.8$ and $M_{\text{inh}} = -9.8$ between the neurons representing the RVs T, C, X and the corresponding auxiliary neurons. The connection strengths between the neurons representing the RVs T, C, B, D and their corresponding auxiliary neurons amount to $M_{\text{exc}} = 9.0$ and $M_{\text{inh}} = -9.0$ due to $\max\left(p(D|T, C, B)\right) = 0.9$. As before, the biases of the principal and auxiliary neurons are calculated via Equations 2.23 and 2.28, respectively. The weights between the principal neurons are calculated according to Equation 2.24.

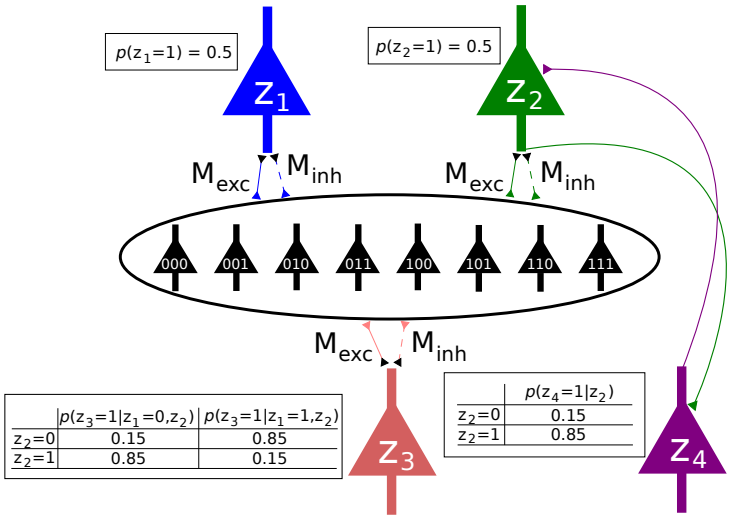


Figure 4.1: Implementation 1 of the visual perception experiment from Figure 2.2. The network contains 8 auxiliary neurons (*black*) in order to represent the third-order factor $p(z_3|z_1, z_2)$.

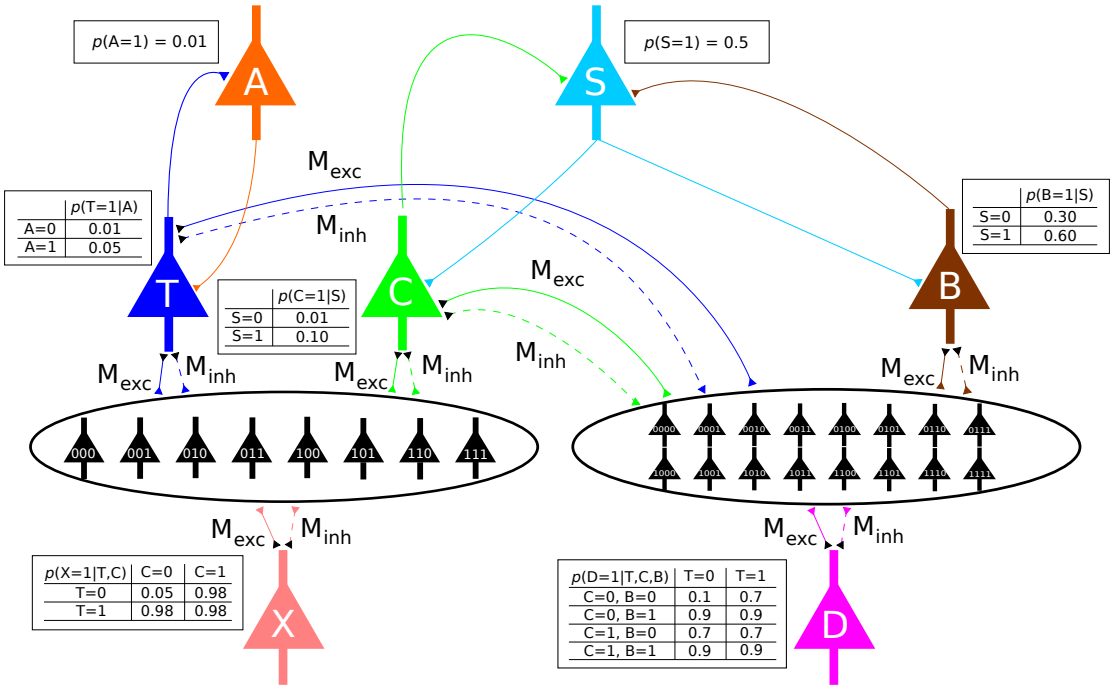


Figure 4.2: Implementation 1 of the ASIA network from Figure 2.3. The network contains 8 auxiliary neurons to express the third-order factor $p(X|T, C)$ and additional 16 auxiliary neurons to represent the fourth-order factor $p(D|T, C, B)$.

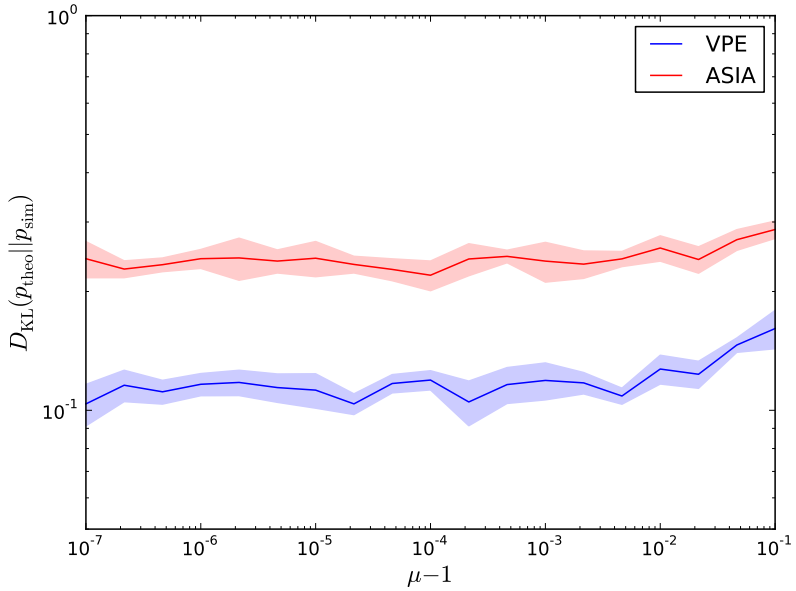


Figure 4.3: Measure of the sampling quality for the VPE and ASIA network with variation of the unspecified parameter μ from Equation 2.28. The duration of a simulation is 100 s. For $\mu = 1 + 10^{-3}$ and lower μ, the KL divergence between the simulated and theoretical joint probability distribution remains constant. For the remainder of this thesis, $\mu = 1 + 10^{-4}$ is chosen.

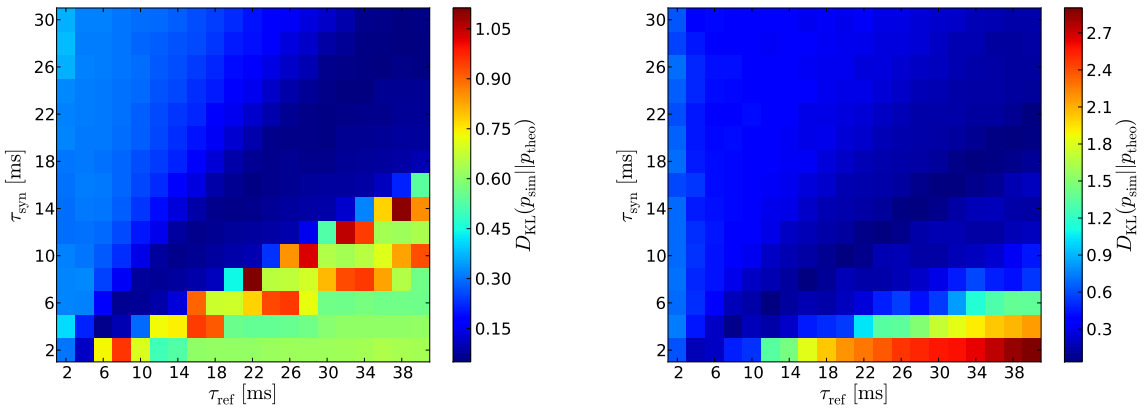
4.2 Parameter Analysis of LIF Neurons for Example Bayesian Networks

A detailed parameter analysis for the two example BNs equivalent to the analysis of Section 3.3 for Boltzmann distributions is in the focus of this section. The following sweeps will yield the LIF neuron parameters which provide for the best sampling results from the BNs described in Section 4.1.

4.2.1 Dependence of LIF Sampling on the Parameter μ

Equation 2.28 introduces the parameter μ which determines the magnitude of the biases of the auxiliary neurons, once they are not inhibited due to the wrong combination of active principal neurons. For example, the bias of the auxiliary neuron which corresponds to the RV X_{001} from the VPE in Figure 4.1 is determined by

$$b_{001} = \log\left(\mu \frac{p\left(z_3 = 1 | z_1 = 0, z_2 = 0\right)}{\min\left[p\left(z_3 | z_1, z_2\right)\right]} - 1\right) - \underbrace{\eta\left(z_1 = 0, z_2 = 0, z_3 = 1\right)}_{=1} M_{\mathrm{exc}} \,. \tag{4.3}$$



A: VPE network **B:** ASIA network

Figure 4.4: Sampling quality for the VPE and ASIA network with variation of the time constants τ_{ref} and τ_{syn}. Each colored square corresponds to the KL divergence of the simulated and theoretical joint probability distribution of all RVs in one simulation of duration $1000\,\mathrm{s} \cdot \tau_{\mathrm{ref}}/30\,\mathrm{ms}$.

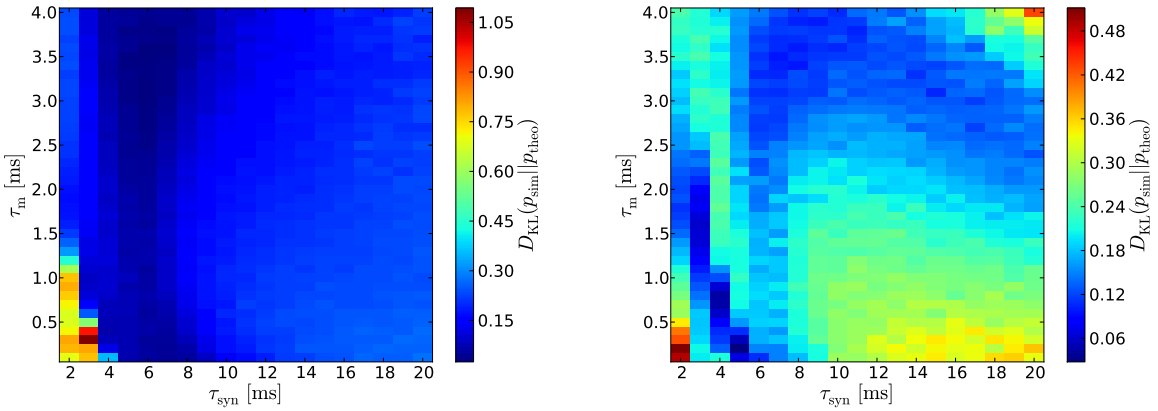
Whenever the principal RVs assume the state $(z_1 = 0, z_2 = 0, z_3 = 1)$, Equation 4.3 reduces to

$$b'_{001} = \log\left(\mu \frac{p\left(z_3 = 1 | z_1 = 0, z_2 = 0\right)}{\min\left[p\left(z_3 | z_1, z_2\right)\right]} - 1\right) , \tag{4.4}$$

since the neuron ν_3 connects with the weight M_{exc} to the auxiliary neuron corresponding to X_{001} and thus the second term of Equation 4.3 vanishes. This allows for a free choice of the parameter μ as long as μ is larger than 1. The closer μ approaches 1, however, the more negative become the values of the biases.
Figure 4.3 illustrates the LIF sampling performance from both BNs described in Section 4.1 depending on the the choice of the parameter μ. Table A.1 lists the parameters of the LIF neurons. For both networks, there is a significant change in sampling performance if μ assumes values larger than $1 + 10^{-3}$. For μ smaller than $1 + 10^{-3}$, the sampling performance stays almost constant. However, for very small μ the absolute values of the biases are very large and the performance may become affected by asymmetrical effects due to the nonlinear impact of PSPs to the course of the membrane potential in the conductance-based LIF models, since then the voltage traces are much closer to $E_{\mathrm{inh}}^{\mathrm{rev}}$ than $E_{\mathrm{exc}}^{\mathrm{rev}}$. For the remaining experiments μ will be fixed to $1 + 10^{-4}$.

4.2.2 Dependence of LIF Sampling from Bayesian Networks on the Parameters of the LIF Neurons

Figure 4.4 shows the LIF sampling results for different choices of the refractory time constant τ_{ref} and the synaptic time constant τ_{syn}. Both BNs entail the similar preference that τ_{syn} has to be slightly smaller than τ_{ref} to achieve the best sampling results. The



A: VPE network **B:** ASIA network

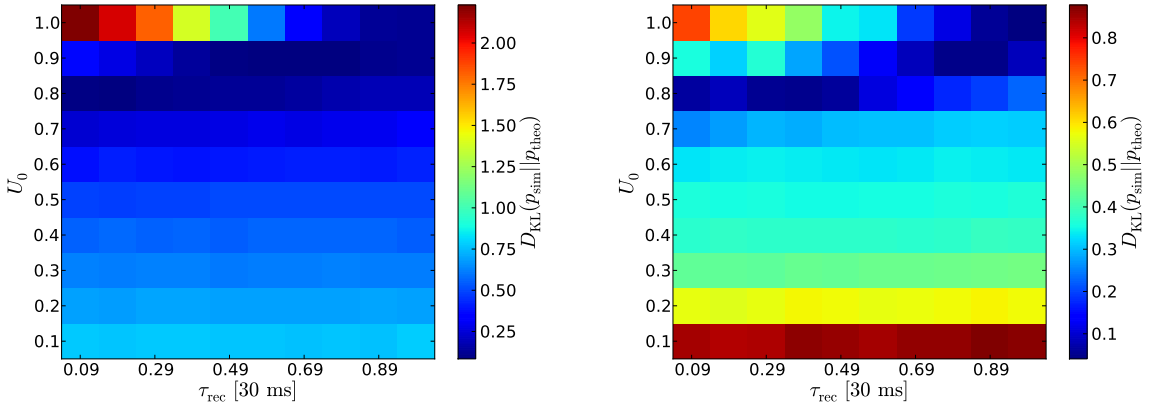
Figure 4.5: Sampling quality for the VPE and ASIA network for different time constants τ_m and τ_{syn}. Each colored square corresponds to the KL divergence of the simulated and theoretical joint probability distribution of all RVs in one simulation of duration 333 s. The parameter τ_{ref} is 10 ms.

best parameters lie for both distributions on the curve $\tau_{syn} \approx 0.5\,\tau_{ref}$. However, the best parameters for the ASIA network have a smaller spread in the parameter space than the best parameters for the VPE. A choice which yields good LIF sampling results from both BNs is $\tau_{syn} = 10$ ms and $\tau_{ref} = 20$ ms.

Figure 4.5 illustrates the LIF sampling quality under variation of the membrane time constant τ_m and the synaptic time constant τ_{syn} with $\tau_{ref} = 10$ ms. Due to the low input frequency of 2×400 Hz, the approximation $\tau_m \approx \tau_{eff}$ is valid. As expected from the discussion in Section 3.3, a very small τ_m and a comparably large τ_{syn} yield good sampling results. Like in Figure 4.4 the parameter choice $\tau_{syn} \approx 0.5\,\tau_{ref}$ is preferred. An interesting observation is that for both BNs the region around $\tau_m = 3.5$ ms and $\tau_{syn} = 6.5$ ms also yield good results. This fact has not been further considered since such large τ_{eff} are unfavorable whenever e.g. a neuron with a large negative bias is perpetually activated by the large synaptic weight from another neuron. The neuron will never reach an ON-state close to $p = 1$ due to the slow membrane dynamics. Another reason is the fact that a small ratio τ_m/τ_{syn} necessary for the validity of the approximation in Equation 2.69.

Figure 4.6 demonstrates the LIF sampling performance for different choices of the STP parameters utilization U_0 and the recovery time constant τ_{rec}. For both BNs the optimal parameter choices are $U = 1$ and $\tau_{rec} \approx \tau_{syn}$ which is a good approximation to renewing PSPs (see Equation 2.77).

Like in Chapter 3, we can conclude that each BN requires its specific parameter choice which achieves the best sampling performance. However, for the two investigated BNs the parameters from Table A.2 yield good sampling results. The preference of $\tau_{syn} < \tau_{ref}$ for networks with large weights can be explained based on Figure 4.7. Due to the shorter



A: VPE network **B:** ASIA network

Figure 4.6: Impact of the STP parameters utilization U_0 and recovery time constant τ_{rec} on the sampling quality for the VPE and ASIA network. The neuron parameters are taken from Table A.2. The duration of each single simulation is 666 s. Each square corresponds to one simulation.

synaptic time constant, the impact of a PSP on the subsequent course of the membrane potential is much smaller than with a large τ_{syn}. This property comes with the fact that the peak height of the PSP is larger compared to the LIF PSP in Figure 3.2 and thus less similar to the ideal rectangular PSP shape. However, a larger peak height compared to Figure 3.2 together with a PSP which has almost completely decayed at τ_{ref} appears to be a good compromise.

4.3 Optimal Results of LIF Sampling from Bayesian Networks

Section 4.2 showed that LIF PSPs with $\tau_{\text{syn}} = 10\,\text{ms}$ and $\tau_{\text{ref}} = 20\,\text{ms}$ allow for good results for sampling from the probability distributions which model the VPE and the ASIA network.

This section offers a comparison of the LIF sampling results with ideal sampling results using stochastic neurons with rectangular or alpha-shaped PSP shapes from the original implementation by *Pecevski et al.* [2011]. Figure 4.7 illustrates the three PSP shapes which all have the refractory time $\tau_{\text{ref}} = 20\,\text{ms}$. In contrast to the PSPs of the stochastic neurons, the LIF PSPs are highly asymmetric and influence the course of the membrane voltage on a larger time scale than τ_{ref}.

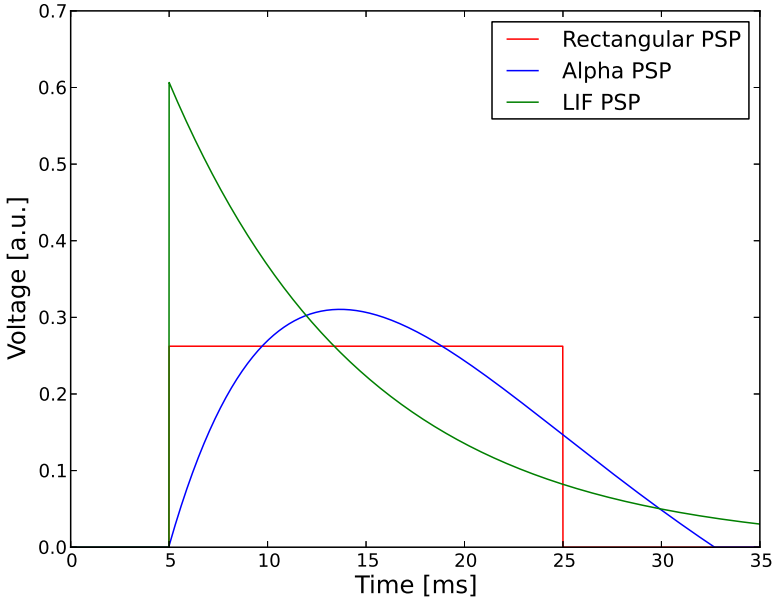
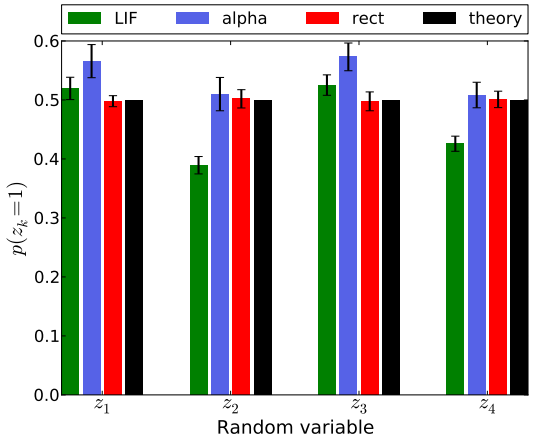


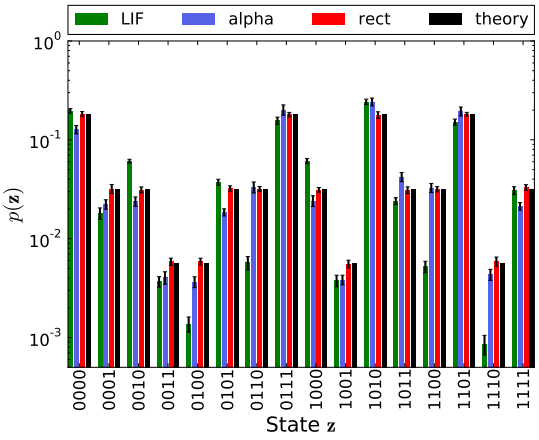
Figure 4.7: The three single neuron PSP shapes with $\tau_{\text{ref}} = 20\,\text{ms}$ which were used in the simulations. The rectangular PSP and alpha-shaped PSP are the original PSPs from *Pecevski et al.* [2011] and are used for neural sampling with stochastic neurons, while the LIF PSPs are used for LIF sampling.

4.3.1 Results of LIF Sampling for the Visual Perception Experiment

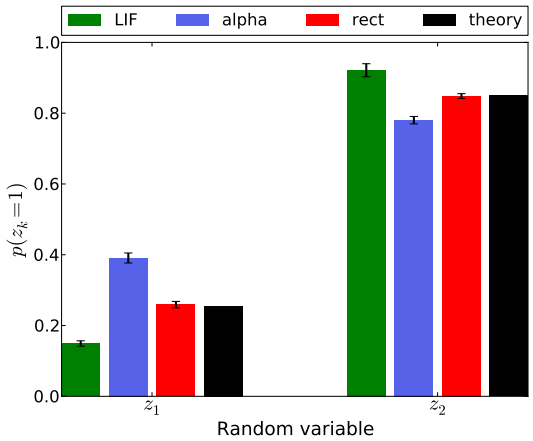
Figure 4.8 shows the sampling results for the VPE averaged over 20 simulations of duration 100 s. The bar charts A and B contain the sampling results of the marginals and joints of the free distribution, C and D the results of sampling from the conditional distribution $p(z_1, z_2|z_3 = 1, z_4 = 1)$, and E and F the results of sampling from the conditional distribution $p(z_1, z_2|z_3 = 1, z_4 = 0)$.
The sampling from these two conditional distributions refers to the inference tasks presented in Section 2.1.3. The conditional probability distribution $p(z_1, z_2|z_3 = 1, z_4 = 1)$ describes the scenario that a sawtooth-like shading profile and cylindrical contour have been observed, which implicates a high probability $p(z_2 = 1)$ that the 3D shape is cylindrical and a high probability $p(z_1 = 0)$ that the relative reflectance is uniform. The distribution $p(z_1, z_2|z_3 = 1, z_4 = 0)$ instead presumes the observation of a flat contour, which results in a high probability $p(z_2 = 0)$ that the 3D shape is flat and a high probability $p(z_1 = 1)$ there is a reflectance step. The clamping of the RVs to 1 or 0 is achieved by injecting a large positive or negative current which is equivalent to setting the bias of the corresponding LIF neurons to 20 or -20, respectively.
For all three sampling tasks the LIF sampling performance is comparable to the results of the stochastic neurons with alpha-shaped PSPs. The average probabilities always lie close to the theoretical values. In contrast to the performance of stochastic neurons with



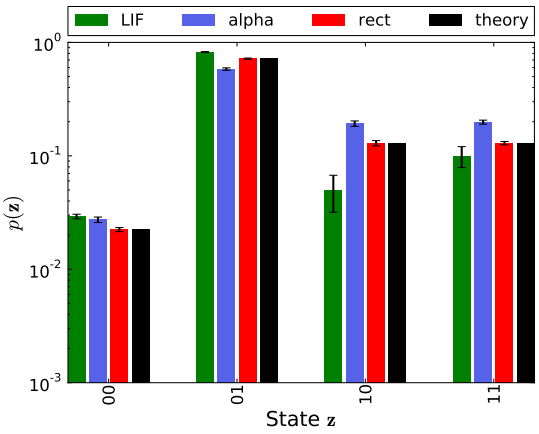
A: Marginals of the free distribution



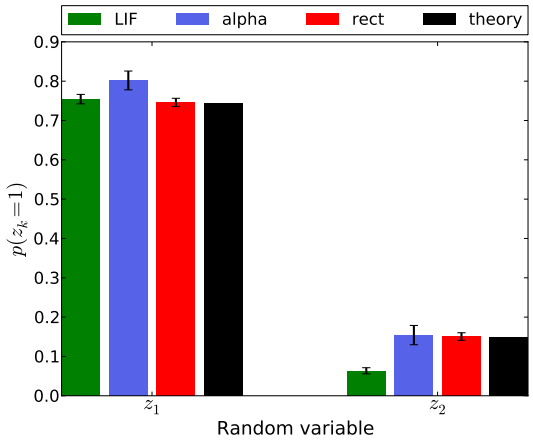
B: Joints of the free distribution



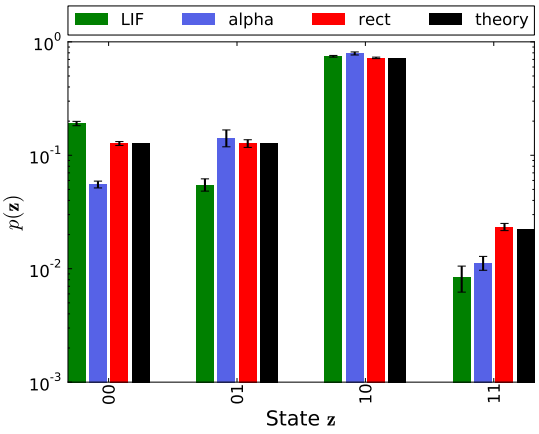
C: Marginals, $\mathbf{e} = (1, 1)$



D: Joints, $\mathbf{e} = (1, 1)$



E: Marginals, $\mathbf{e} = (1, 0)$



F: Joints, $\mathbf{e} = (1, 0)$

Figure 4.8: Comparison of the sampling performance for simulations of Implementation 1 with LIF PSPs with parameters from Table A.2, alpha-shaped PSPs, ideal rectangular PSPs and the target probabilities for the VPE. The left column shows the marginal distributions and the right column the corresponding joint probability distributions of the random variables z_1 and z_2. The evidence $\mathbf{e}$ corresponds to the vector (z_3, z_4). The bars show the average results of 20 simulations each of duration 100 s. The error bars denote the standard deviations.

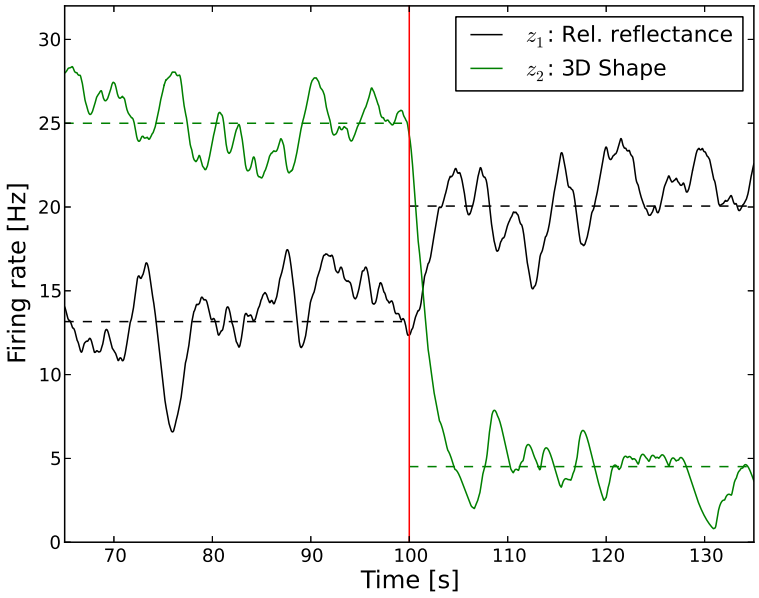


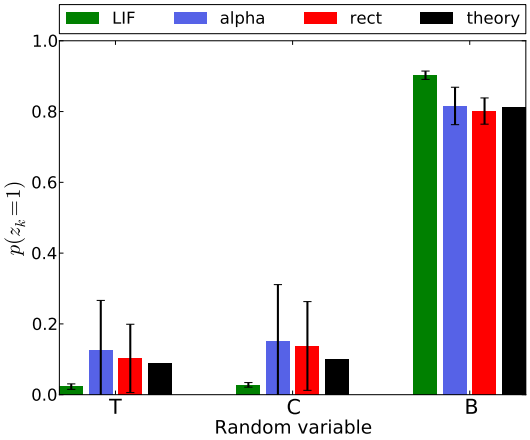
Figure 4.9: The firing rates of the principal neurons ν_1 and ν_2 from the VPE. The rate was determined by convolving the spike pattern with a kernel $K(t) = \frac{t}{\tau} \exp\left(-\frac{t}{\tau}\right)$ with $\tau = 0.1\,\text{s}$. At time point 100 s (red line), the evidence was switched from $\mathbf{e} = (z_3 = 1, z_4 = 1)$ to $\mathbf{e} = (z_3 = 1, z_4 = 0)$. The evolution of the firing rates demonstrates the "explaining away" effect. The *dashed* lines denote the mean firing rates.

ideal rectangular PSPs the average probabilities of the LIF sampling results differ more than one standard deviation from the theoretical probabilities in most cases.
The LIF sampling results of the marginals of the free distribution imply that the two directly connected neurons ν_2 and ν_4 (see Figure 4.1) spike too rarely. This might arise from the fact that the weights between ν_2 and ν_4 are much smaller than the weights between ν_1, ν_2, ν_3 and the auxiliary neurons. Thus the network activity is shifted away from the interactions of the two neurons.
The LIF results of the two inference tasks have one characteristic. If the theoretical value is lower than $p\,(z = 0.5)$, the LIF sampling result is below the theoretical value. If the theoretical value is larger than $p\,(z = 0.5)$, the LIF sampling result is above the theoretical value. This aspect might happen for several reasons. First, it is a hint for too large peaks of the synaptic PSPs, which lead to the fact that neurons with a bias close to 0 are forced to spike more often than expected by the synaptic inputs from other neurons. Second, due to the non-vanishing influence of a PSPs on the course of the membrane voltage beyond τ_{ref} e.g. neurons which tend to spike rarely are even longer inhibited than expected. Third, the small deviations of the activation function from an ideal logistic function could lead to lower or larger biases than expected, respectively (see Figure 3.1).

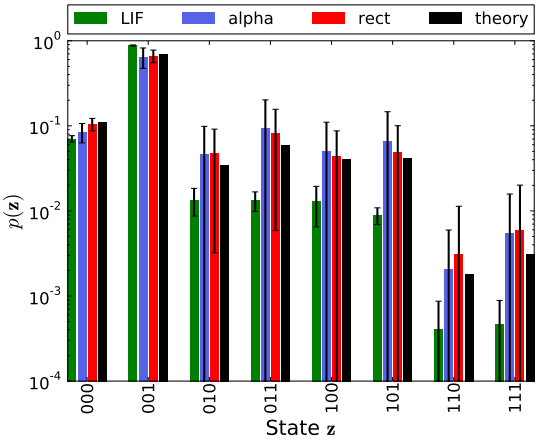
Figure 4.9 shows the evaluation of the firing rates of the neurons ν_1 and ν_2 for the two inference tasks. During the first 100 s the RVs z_3 and z_4 are clamped to 1. In this case, the neuron ν_1 has a firing rate of about 13 Hz and the neuron ν_2 of about 25 Hz. This situation corresponds to the inference task in Figure 4.8C and D. At time point 100 s, the RV z_4 is clamped to 0. Now we have the inference situation of Figure 4.8E and F. After a transition of about 5 s, the firing rates of the neurons ν_1 and ν_2 settle to new values and remain there in average. Now, the neuron ν_1 elicits spikes with a frequency of about 20 Hz and the neuron ν_2 with about 4 Hz. These values correlate with the probabilities which were inferred in Figure 4.8.

4.3.2 Results of LIF Sampling from the ASIA Network

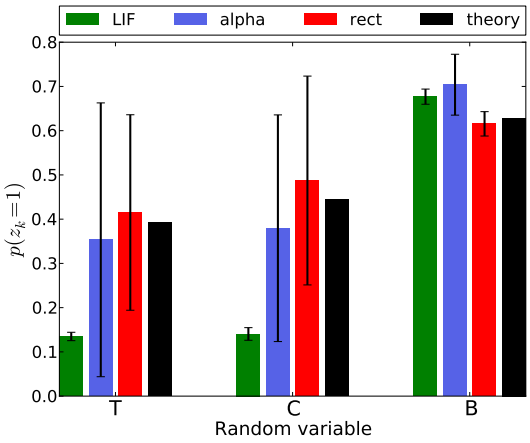
The same performance comparison as in Section 4.3.1 was provided for the ASIA network. Here the probabilities of a patient having one of the three diseases tuberculosis (T), lung cancer (C) and bronchitis (B) are inferred for two different situations. Figures 4.10A and B display the results of sampling from the conditional probability distribution $p(T, C, B|A = 1, D = 1)$: this distribution models the situation in which the patient has recently visited Asia ($A = 1$) and suffers from dyspnoea ($D = 1$). The diagnosis that the person has a bronchitis is much more probable than any of the the other diseases. Figures 4.10C and D present the results of sampling from the conditional probability distribution $p(T, C, B|A = 1, X = 1, D = 1)$. In this case, an additional X-ray test has been accomplished with a positive result. Now the diagnosis that the person has a bronchitis is not as certain any more and the probabilities to have any of the other diseases are almost as large as the probability to have bronchitis.
In both cases, the results of LIF sampling from the conditional probability distributions are unsatisfactory compared to the sampling results with stochastic neurons. The sampled probabilities with LIF neurons always differ more than two standard deviations from the corresponding theoretical values. In particular, Figure 4.10C implies that the LIF neurons fall short of sampling correctly from the distribution $p(T, C, B|A = 1, X = 1, D = 1)$. According to Figure 4.10D, the states $(T = 0, C = 0, B = 0)$ and $(T = 0, C = 0, B = 1)$ are occupied systematically more frequent than statistical fluctuations would account for at an expense of the occurrence of states $(T = 0, C = 1, B = 1)$ and $(T = 1, C = 0, B = 1)$. The unsatisfactory LIF sampling results can be mainly traced back to the large differences of weights and biases induced by the extreme values of the conditionals $p(X = 1|T, C)$ (see Figure 4.2). Furthermore, both RVs T and C interact with two groups of auxiliary RVs, while the RV B is only influenced by one group of auxiliary RVs. This might lead to the fact that the neurons associated with T and C are e.g. longer inhibited because the PSP impact on the course of the membrane voltage is nonzero beyond τ_{ref} and both neurons are inhibited by large negative weights from two groups of auxiliary neurons.
The standard deviations of the sampling results of the stochastic neurons also express the difficulty to sample from the given conditional probability distributions of the ASIA network. With much longer simulation times, the results can be vastly improved. However, this can not be applied to the LIF sampling results. Here, the standard deviations are already very small and longer simulation runs would not improve the sampling performance,



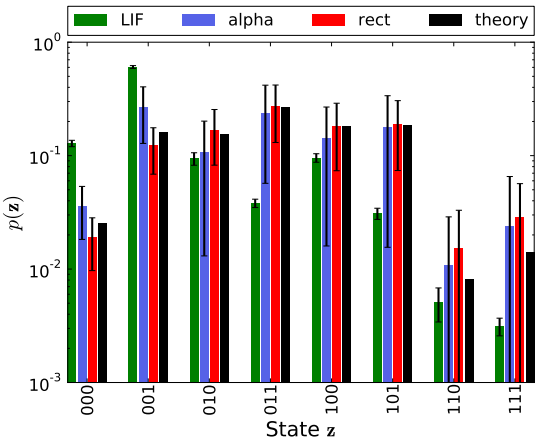
A: Marginals, $\mathbf{e} = (A = 1, D = 1)$



B: Joints, $\mathbf{e} = (A = 1, D = 1)$



C: Marginals, $\mathbf{e} = (A = 1, X = 1, D = 1)$



D: Joints, $\mathbf{e} = (A = 1, X = 1, D = 1)$

Figure 4.10: Comparison of the sampling performance for simulations of Implementation 1 with LIF PSPs with parameters from Table A.2, alpha-shaped PSPs, ideal rectangular PSPs and the target probabilities for the ASIA network. The left column shows the marginal distributions and the right column the corresponding joint probability distributions of the three RVs of interest: T, C and B. The bars show the average results of 20 simulations of duration 100 s. The error bars denote the standard deviations.

which implies systematic mismatches. An eligible question at this point is whether there is an option to modify the LIF PSP shape towards the optimal rectangular PSP shape. The next section will introduce mLIF PSP with the help of which the sampling results will improve considerably.

4.4 LIF Sampling Improvement via mLIF PSPs

In the course of the *4th BrainScaleS Demo3 workshop* [2013], the idea originated that a LIF PSP shape which is closer to an ideal rectangular shape and which abruptly vanishes after the refractory time τ_{ref} will drastically improve the LIF sampling results. A more rectangular LIF PSP shape would mimic the ideal PSP shape of stochastic neurons and comprise the advantage of being symmetrical. A PSP shape which vanishes after the refractory time would have the advantage that it will not influence the course of the membrane potential of the neuron beyond τ_{ref}.
In the following, first the generation of such a so-called mLIF[1] PSP shape is described. Afterwards, the LIF sampling results with the new mLIF PSPs are compared to the results with the standard LIF PSPs. In the end of this section a temporal analysis of the convergence behavior with LIF PSPs and mLIF PSPs is presented.

4.4.1 Engineering mLIF PSPs

The generation of a PSP shape with LIF neurons which is close to the ideal rectangular shape requires the interplay of the dynamics of multiple neurons for at least two reasons. First, a single neural membrane with the requirement of a fast rising edge of the membrane potential will result in a highly asymmetrical shape. Thus the falling edge has to be prolonged such that the membrane stays at an almost constant potential during the refractory period. Second, a prolongation of the PSP shape will even increase the size of the PSP at the end of the refractory period and thus increase the impact on the subsequent course of the membrane potential. For this reason, an additional neuron is needed, which abruptly terminates the PSP.
Figure 4.11 illustrates the general idea to design such a PSP shape with LIF neurons, which will be referred to as mLIF PSP. One principal or auxiliary sampling neuron of the original Implementation 1 is replaced by a group of neurons. The first neuron remains the actual sampling neuron with bias and weights as introduced in Section 2.1.5. The remaining neurons are parrot neurons which forward the action potentials of the sampling neuron. In the following I will refer to the first neuron as *sampling neuron* and to the remaining neurons as *forwarding neurons*.
In addition to the existing network connectivity, each of the forwarding neurons projects onto the same sampling neurons to which their associated sampling neuron is connected. The biases of the forwarding neurons and the weights to the forwarding neurons are selected such that a forwarding neuron is directly triggered to spike upon each incoming spike. The last of the forwarding neurons connects with the opposite sign to postsynaptic

[1] The letter "m" in *mLIF* refers to "multiple" since we are now using *multiple* neurons per RV.

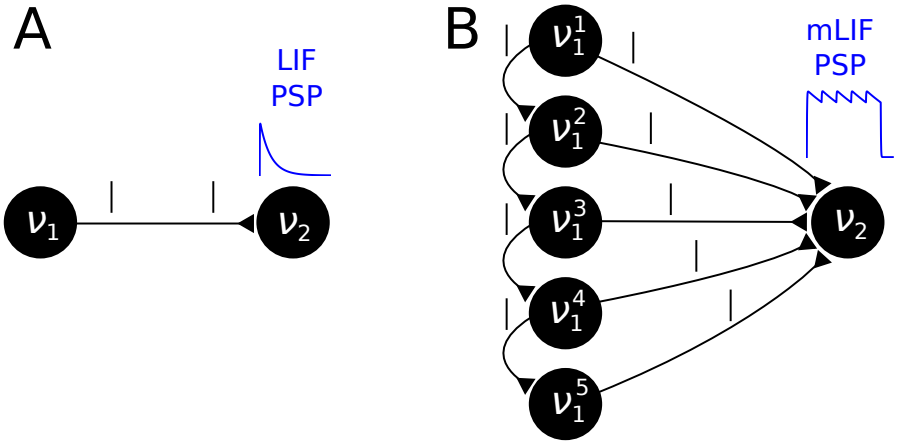
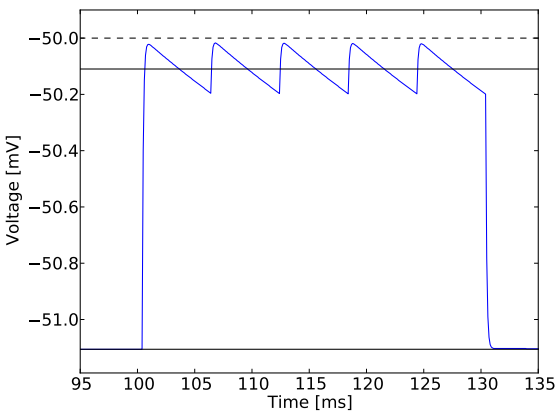
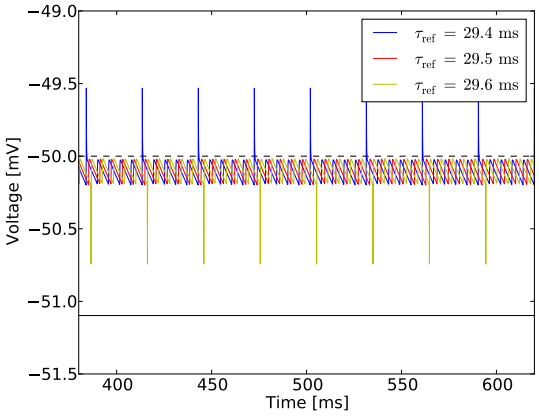


Figure 4.11: The previous experiments were simulated with a standard LIF PSP shape **(A)**. To establish a PSP shape which is closer to the ideal rectangular shape, the following network structure was set up **(B)**: Instead of using one principal neuron ν for one random variable, each random variable is represented by a chain of neurons of which the first neuron is the actual sampling neuron. Additionally to the network connections from Figure 4.1, there are feed forward connections along this neural chain. Each of the chain neurons connects to the postsynaptic sampling neuron (all connections from ν_1^i to ν_2) which itself is the sampling neuron of another feed forward chain. By choosing appropriate synaptic efficacies, a sawtooth like PSP shape can be artificially established which is close to the desired rectangular shape.

sampling neuron. This allows to directly terminate the PSP on the postsynaptic site. This last forwarding neuron automatically provides for renewal PSPs even without the presence of STP.
One will ensure a sawtooth-like PSP shape which is shown in Figure 4.12A, if the delays and the weights connecting the forwarding neurons and also those connecting the forwarding neurons and the following sampling neurons are chosen deliberately. Table A.4 lists the parameters of the extended neural network.
Several issues when engineering mLIF PSP shapes need to be considered. First, the size of the teeth of the PSP shape has to be constructed such that it is smaller than the overall size of the PSP. This prevents from strong variation of the firing probability of the postsynaptic neuron during the refractory period of the presynaptic sampling neuron. A second issue is illustrated in Figure 4.12B. The delays between the forwarding neurons have to be chosen such that once a sampling neuron ν_1 is clamped to a very high membrane potential, the postsynaptic sampling neuron ν_2 will receive a constant offset. An incautious selection of the delays will result in terminating and directly arising PSPs after each refractory period and might cause undesired threshold crossings.
The same PSP shape could also be modeled in a different manner. This involves the simultaneous activation of the forwarding neurons by the sampling neuron. Due to different synaptic weights and membrane time constants, the time course of the postsynaptic action



A: Resulting *mLIF* PSP shape by applying the method described in Figure 4.11.

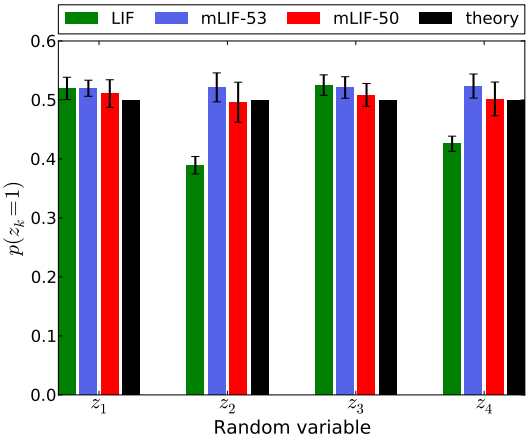


B: Membrane potential of a noise-free LIF neuron with a constant offset

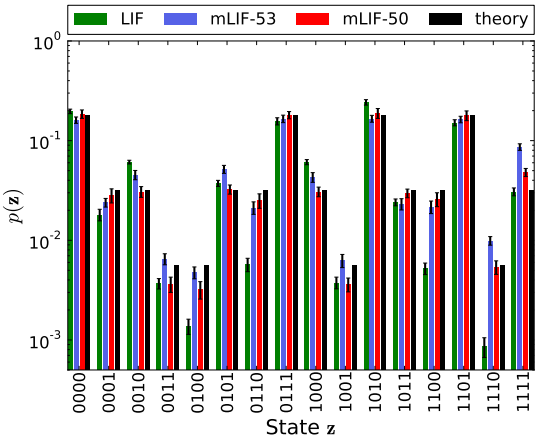
Figure 4.12: Panel **A** shows the resulting *mLIF* PSP shape of the method described in Figure 4.11. The neural chains which are used in this chapter consist of 6 neurons: one neuron to initiate the PSP, 4 neurons to maintain the level of the membrane potential and one neuron to cancel the PSP. Table A.4 lists the parameters of this extended neural network. Panel **B** shows the criterion which was used to find the optimal refractory time τ_{ref} for a given number of chain neurons: a sampling neuron receives the spiking input of a presynaptic sampling neuron and its forwarding neurons. Without Poisson background input the membrane just receives an almost constant offset current, which is the stringing together of consecutive mLIF PSPs. A refractory time of $\tau_{\text{ref}} = 29.5\,\text{ms}$ is preferred for 6 neurons and $\tau_{\text{syn}} = 30\,\text{ms}$. With $\tau_{\text{ref}} = 29.4\,\text{ms}$ or $\tau_{\text{ref}} = 29.6\,\text{ms}$, there are potential jumps after each τ_{ref} which may cause undesirable threshold crossings.

potential of each forwarding neuron would be delayed for each neuron differently which would result in the same PSP course on the membrane of the postsynaptic sampling neuron as shown in Figure 4.12. However, there are several crucial disadvantages compared to the method explained before. First, this method would require an individual adjustment of the membrane time constants for each neuron which becomes very exhausting for large groups of neurons and long refractory times. Second, this method would require an arbitrary flexibility and a high precision of the membrane time constants. This will not be compatible with the inherent variations of a neuromorphic hardware device (see Chapter 6).

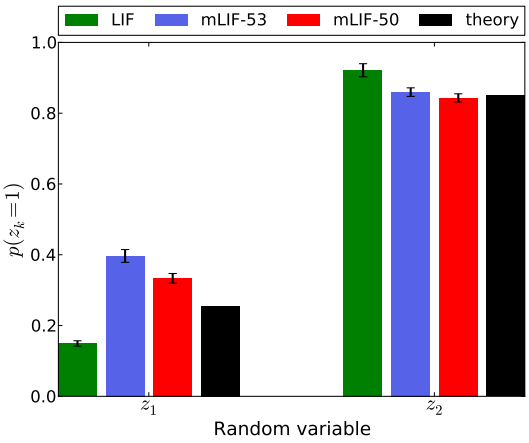
The next sections will compare the sampling performance of sampling with LIF PSPs and mLIF PSPs for the VPE and the ASIA network. It will become apparent that the mLIF PSPs outperform the LIF PSPs by a large margin.



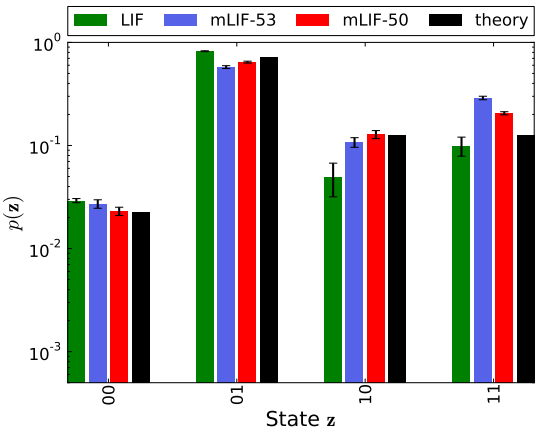
A: Marginals of the free distribution



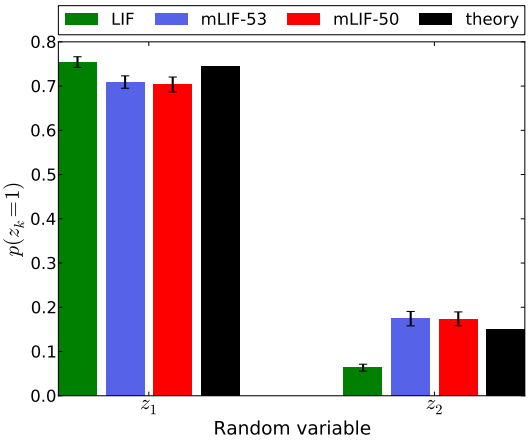
B: Joints of the free distribution



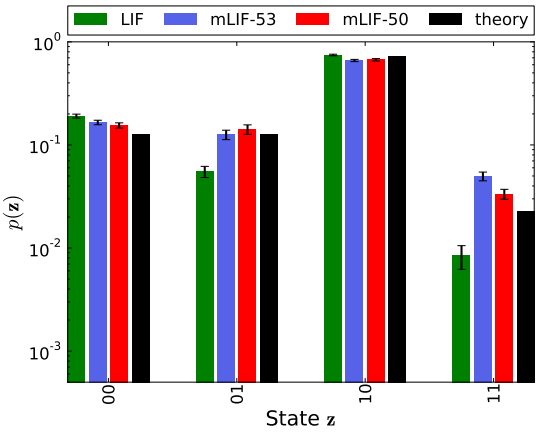
C: Marginals, $\mathbf{e} = (1, 1)$



D: Joints, $\mathbf{e} = (1, 1)$



E: Marginals, $\mathbf{e} = (1, 0)$



F: Joints, $\mathbf{e} = (1, 0)$

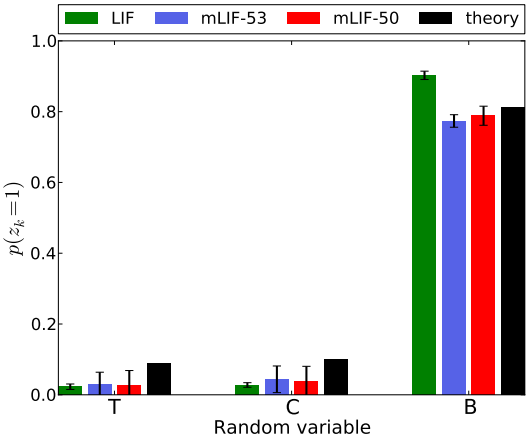
Figure 4.13: Comparison of the LIF sampling performance via Implementation 1 with standard LIF neurons with parameters from Table A.2, two versions of Implementation 1 with the improved PSPs from Figure 4.11B and the target probabilities for the VPE. The versions mLIF_{-50} and mLIF_{-53} have different reset voltages -50.01 mV and -53.0 mV, respectively. The left column contains the marginal distributions and the right column the corresponding joint probability distributions of the random variables z_1 and z_2. The evidence $\mathbf{e}$ corresponds to the vector (z_3, z_4). The bars show the average results of 20 simulations of duration 100 s for the LIF PSP, 150 s for the mLIF PSPs. The error bars denote the standard deviations.

4.4.2 Results of LIF Sampling with mLIF PSPs for the Visual Perception Experiment

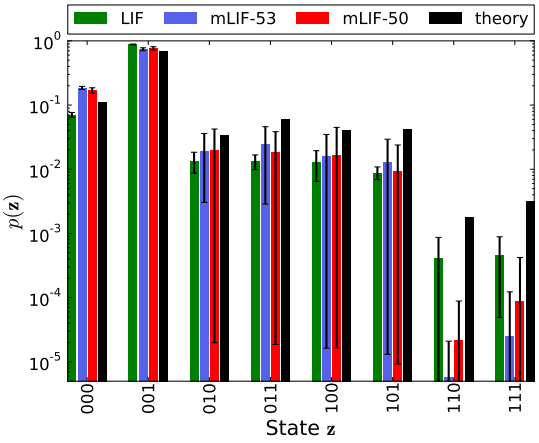
Two different versions of mLIF PSPs were used to test the sampling performance and compare it to the performance with standard LIF PSPs. A.2 lists the parameters of the neurons with standard LIF PSPs. The parameters for generating the mLIF PSPs are taken from Table A.4. The PSPs mLIF_{-50} and mLIF_{-53} only differ in the reset potential of the sampling neuron. mLIF_{-50} refers to an optimal reset potential of $V_{\text{reset}} = -50.01\,\text{mV}$ which is close to the threshold potential such that the neuron is allowed to spike again directly after the refractory period. mLIF_{-53} refers to the reset potential $V_{\text{reset}} = -53.0\,\text{mV}$ which e.g. takes account for the compatibility with the fluctuations of floating gate voltages and currents on the FACETS Wafer-Scale Hardware System, with respect to a prospective implementation of mLIF PSPs. Due to the reset potential $V_{\text{reset}} = -53.0\,\text{mV}$, the neuron needs a longer time to recover to its free membrane potential after it has passed the refractory time, which shifts neural correlations and, therefore, leads to a slight imprecision of the sampling results compared to $V_{\text{reset}} = -50.01\,\text{mV}$.
Figure 4.13 shows the LIF sampling results for the VPE of 20 simulation runs of 100.0 s each for LIF PSPs and of 150.0 s each for mLIF PSPs. This distinction is made due to different refractory times τ_{ref}. For each of the three probability distributions $p(z_1, z_2, z_3, z_4)$, $p(z_1, z_2|z_3 = 1, z_4 = 1)$ and $p(z_1, z_2|z_3 = 1, z_4 = 0)$, the sampling results with mLIF PSPs are superior. The theoretical probabilities are mostly in the range of two standard deviations of the simulated probabilities for both, mLIF_{-50} and mLIF_{-53}. Only for the distribution $p(z_1, z_2|z_3 = 1, z_4 = 1)$ (see Figure 4.13C and D) larger deviations are observable, but which still the results are better than the results obtained via LIF PSPs.

4.4.3 Results of LIF Sampling with mLIF PSPs from the ASIA Network

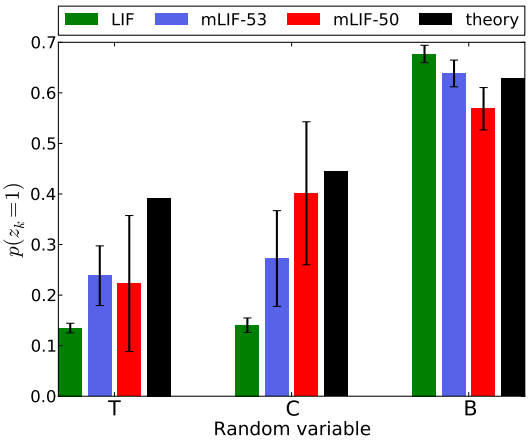
Figure 4.13 shows the LIF sampling results for the ASIA network of 20 simulation runs of 100.0 s each for LIF PSPs and of 150.0 s each for mLIF PSPs. With both implementations of mLIF PSPs, the LIF sampling quality distinctly improves for both probability distributions $p(T, C, B|A = 1, D = 1)$ and $p(T, C, B|A = 1, X = 1, D = 1)$. The theoretical probabilities are always in the range of two standard deviations of the simulated probabilities for both versions of the mLIF PSP.
The most distinctive result is the sampling performance for the distribution $p(T, C, B|A = 1, X = 1, D = 1)$ in Figure 4.13C and D. Similar to the results with stochastic neurons in Figure 4.10 the standard deviations of the results of the simulations with mLIF PSP become very large. This might indicate two facts. On the one hand, the convergence of Implementation 1 is known to be very slow due to the additional set of auxiliary RVs [*Levin et al.*, 2006]. On the other hand, the underlying Bayesian network contains prior and conditional probabilities which are close to 0 or 1 and thus lead to large variations of firing frequencies throughout the network.
The following section offers an analysis of the convergence time of LIF sampling with LIF and mLIF PSPs based on the presented example BNs.



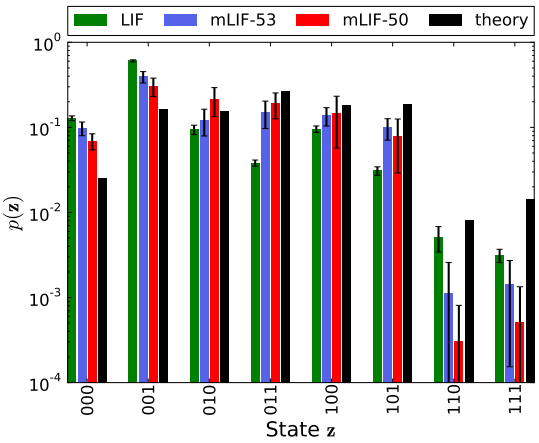
A: Marginals, $\mathbf{e} = (A = 1, D = 1)$



B: Joints, $\mathbf{e} = (A = 1, D = 1)$



C: Marginals, $\mathbf{e} = (A = 1, X = 1, D = 1)$



D: Joints, $\mathbf{e} = (A = 1, X = 1, D = 1)$

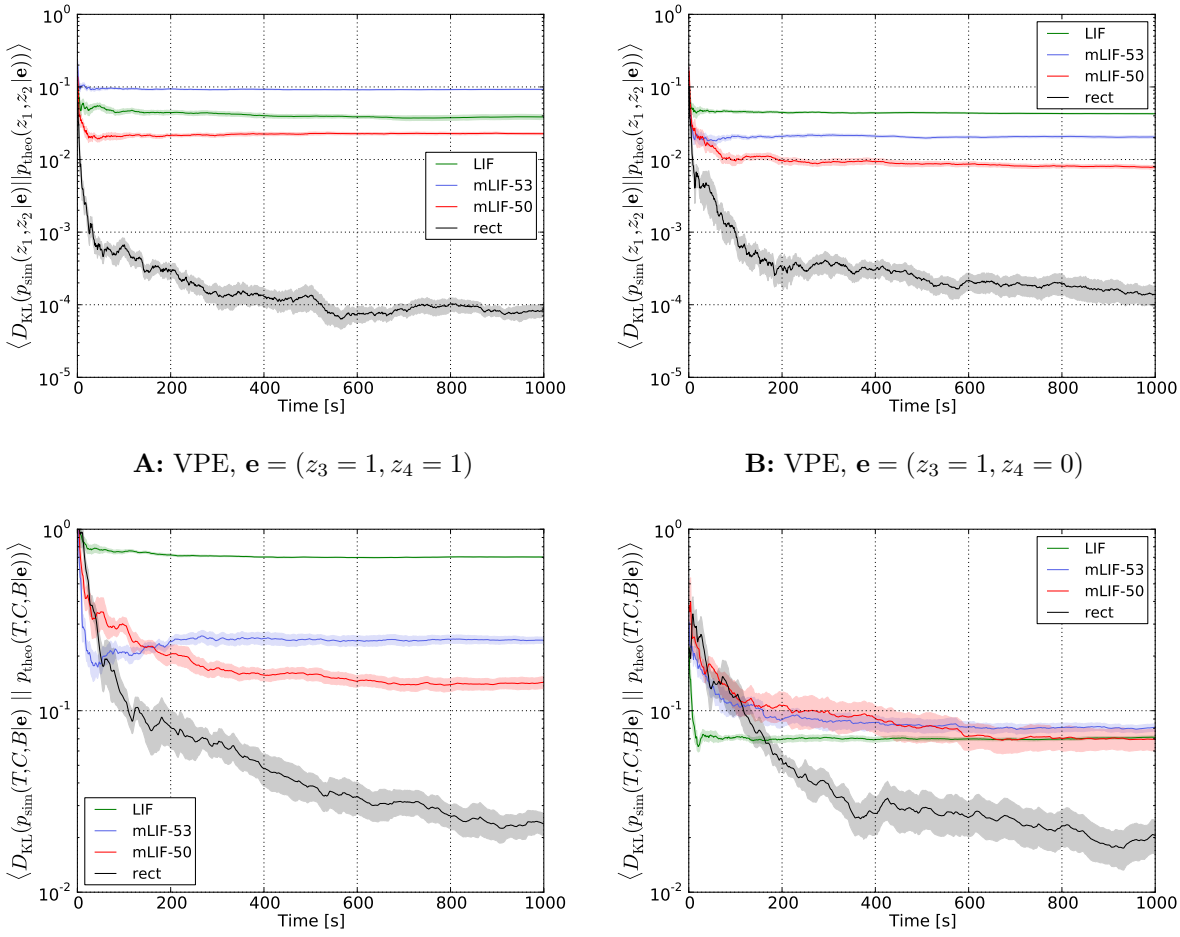
Figure 4.14: Comparison of the LIF sampling performance via Implementation 1 with standard LIF neurons with parameters from Table A.2, two versions of Implementation 1 with the improved PSPs from Figure 4.11B, and the target probabilities for the ASIA network. The versions mLIF$_{-50}$ and mLIF$_{-53}$ have different reset voltages -50.01 mV and -53.0 mV, respectively. The left column contains the marginal distributions and the right column the corresponding joint probability distributions of the random variables T, C and B. The bars show the average results of 20 simulations of duration 100 s for the LIF PSP, 150 s for the mLIF PSPs. The error bars denote the standard deviations.

4.4.4 Temporal Evolution of the KL Divergence

Figure 4.14 already indicated that the convergence of the sampled probability distribution towards the stationary probability distribution is very slow for mLIF PSPs. In the following, the convergence times of sampling with LIF PSPs and mLIF PSPs are compared. As before, Table A.2 lists the parameters for LIF sampling with LIF PSPs. Table A.4 contains the parameters for the LIF sampling with mLIF PSPs.
Figure 4.15 presents the temporal evolution of the KL divergence between the simulated and the target joint probability distributions for the VPE, in Panels A and B, and the ASIA network, in Panels C and D. Panels A and B refer to the results of sampling from the distributions $p(z_1|z_3 = 1, z_4 = 1)$ and $p(z_1|z_3 = 1, z_4 = 0)$ of the VPE, respectively. Panels C and D show the results of sampling from the distributions $p(T, C, B|A = 1, X = 1, D = 1)$ and $p(T, C, B|A = 1, D = 1)$ of the ASIA network, respectively.
For all four probability distributions, the convergence time of sampling with LIF PSPs is in the range of 10 s. With mLIF PSPs the convergence time is about 100 s for the VPE and larger than 100 s for the ASIA network. This can be partly explained by the fact that he refractory time constant is longer for mLIF PSPs than for LIF PSPs. Another reason is that for mLIF PSPs the firing times of the sampling neurons are more distributed along the duration of the PSP. Implementation 1 rather favors PSP shapes for which the firing occurs directly after activation ("all or nothing") and which better mimic ideal infinite weights (see Section 2.1.5). With regard to the convergence time, this feature fits better to the asymmetric LIF PSPs because they peak at the beginning of τ_{ref}.
However, the slow convergence and longer simulation times due to a larger number of neurons comes with the advantage of a more accurate sampling result which militates for the implementation of mLIF PSPs. The obviously better sampling performance with LIF PSP in Panel A and D of Figure 4.15 is misleading. If comparing with the actual sampling results in Figures 4.13C and 4.14C, respectively, the results of LIF sampling with mLIF PSPs are better. This discrepancy is an artifact of the asymmetry of the KL divergence (see Equation 2.15).

4.5 Implementation 1: LIF Sampling Performance on General Bayesian Networks

So far, the sampling performance via Implementation 1 via LIF neurons has been tested on the two specific BNs presented in Section 4.1. As for sampling from BMs in Chapter 3, it would be desirable to gain a more general result of LIF sampling from BNs and compare sampling with LIF PSPs to sampling with mLIF PSPs from random BNs. For this, Section 4.5.1 describes a procedure to create random BNs with pairwise random prior and conditional probabilities. Section 4.5.2 presents the results of sampling from random BNs of 5 RVs with LIF neurons and compares them to the sampling results of stochastic neurons with ideal PSPs.



A: VPE, $\mathbf{e} = (z_3 = 1, z_4 = 1)$

B: VPE, $\mathbf{e} = (z_3 = 1, z_4 = 0)$

C: ASIA, $\mathbf{e} = (A = 1, X = 1, D = 1)$

D: ASIA, $\mathbf{e} = (A = 1, D = 1)$

Figure 4.15: Temporal evolution of the mean KL divergence of the simulated and target joint probability distribution of the RVs z_1 and z_2 from the VPE (*upper row*) and of the RVs T, C and B from the ASIA network (*lower row*) for the three PSP shapes LIF, mLIF$_{-50}$, mLIF$_{-53}$ as well as the ideal rectangular PSP shape. For each plot, the given evidence $\mathbf{e}$ is different. Each curve denotes the average result of 10 simulations of duration 1000 s. The shaded regions denote the standard error of the mean.

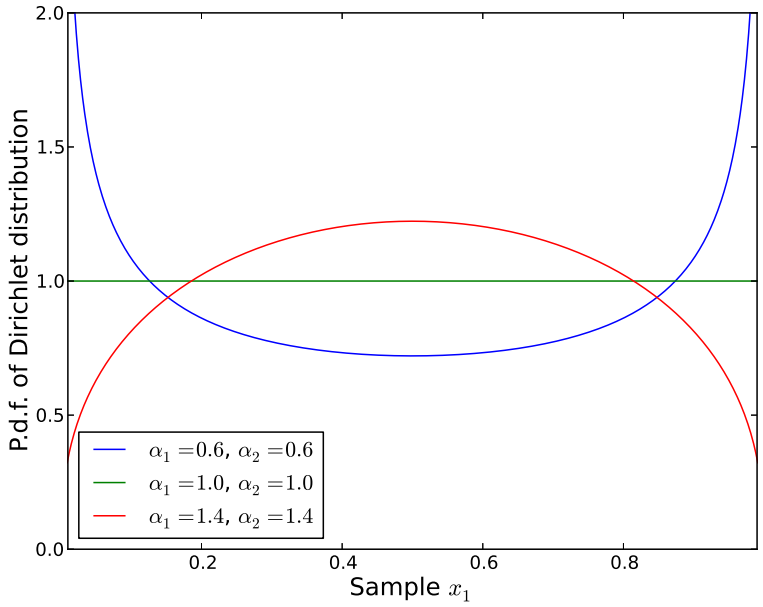


Figure 4.16: Symmetrical two-dimensional Dirichlet distributions for different parameters α_1=α_2=α. The prior and conditional probabilities of the randomly generated Bayesian networks are samples from these Dirichlet distributions. The parameter α allows to choose the extreme nature of the resulting BN. The smaller α is, the larger is the probability to choose a sample close to the boundaries 0 or 1. With higher α, the discrepancy of the prior and conditional probabilities decreases.

4.5.1 Generating General Bayesian Networks

Ide and Cozman [2002] present a method to generate random BNs of K binary RVs $z_1, z_2, ..., z_K$ and pairwise random prior and conditional probabilities. The algorithm starts with the simplest possible connectivity pattern for which a BN is fully connected, which is a Markov chain (see Section 2.1.4): z_1 is connected to z_2, z_2 is connected to z_3 and so on. The term *fully connected* means that irrespective of the direction of the edges in the BNs every node can be reached from every other node. The algorithm runs for N iterations and in each iteration randomly creates a pair (z_i, z_j) of RVs with the condition $z_i < z_j$. For each pair one of the two following actions is performed:

- If the connection $z_i \to z_j$ exists, it is removed, but only if the BN remains fully connected. Otherwise nothing is done.
- If the connection $z_i \to z_j$ does not exist, then it is added, but only if z_i and z_j have less than 8 connections to other nodes. Otherwise nothing is done.

This algorithm leads to a BN with a random connectivity pattern.

The BN then needs to be fully defined by assigning prior and conditional probabilities similar to these of the VPE and the ASIA network in Figure 4.1 and 4.2. For this, the prior and conditional probabilities x_i are randomly drawn from a Dirichlet distribution of second order

$$D\left(x_1, x_2, \alpha_1, \alpha_2\right) = \frac{1}{B\left(\alpha\right)} \prod_{i=1}^{2} x_i^{\alpha_i - 1} \,, \tag{4.5}$$

with the multinomial Beta function

$$B\left(\alpha\right) = \frac{\prod_{i=1}^{2} \Gamma\left(\alpha_i\right)}{\Gamma\left(\sum_{i=1}^{2} \alpha_i\right)} \,, \tag{4.6}$$

with the Gamma function $\Gamma\left(\alpha\right)$, the vector $\alpha = \left(\alpha_1, \alpha_2\right)$ and the condition $x_1 + x_2 = 1$. The last condition means that the probabilities x_i are pairwise determined: once x_1 has been drawn, x_2 denotes $1 - x_1$. Choosing the parameters $\alpha_1 = \alpha_2 := \alpha$ makes the Dirichlet distribution symmetrical.
Figure 4.16 shows the probability density function of a Dirichlet distribution as a function of α. The case $\alpha = 1$ is the one-dimensional uniform distribution. For $\alpha < 1$, the probability to draw samples close to the boundaries 0 and 1 is larger than the probability to draw samples from the middle of the interval. For $\alpha > 1$, the probability to draw samples around 0.5 is larger than the probability to draw samples close to the boundaries 0 and 1. By choosing α, we determine the extreme nature of the resulting BN.

4.5.2 Performance Comparison of Sampling from General Bayesian Networks

The algorithm to generate random BNs described in Section 4.5.1 is applied for BNs of $K = 5$ RVs and $N = 50000$ iterations. The parameter α is varied between 0.3 and 10. For each α, 10 random BNs are created. For each of these BNs, the sampling performance with standard LIF PSPs and mLIF PSPs is measured and compared to the performance of stochastic neurons with ideal rectangular PSPs. The duration of one simulation is 500 s for all investigated PSPs.
Figure 4.17 illustrates the average sampling results for the different PSP shapes as a function of α. For larger α, the sampling performance increases for each of the PSPs after the same simulation time. This can be explained by the fact that a large α favors prior and conditional probabilities which are rather close to 0.5 than to the boundaries 0 and 1. In this case, we have a smaller discrepancy of the prior and conditional probabilities, and, thus, a smaller discrepancy of the weights and biases in the neural representation of the BNs due to Equations 2.27 and 2.28. It has been shown in Figure 3.3 that sampling from BMs with low weights and biases yields better results after the same simulation time.
Sampling with mLIF PSPs leads to results which are up to one order of magnitude more precise than sampling with LIF PSPs in terms of KL divergence between the simulated and the target joint probability distribution. Among the mLIF PSPs, sampling with mLIF_{-50} PSPs is more precise than sampling with mLIF_{-53} PSPs for each α, a consequence of the different choice of the reset potential (see Section 4.4.2).

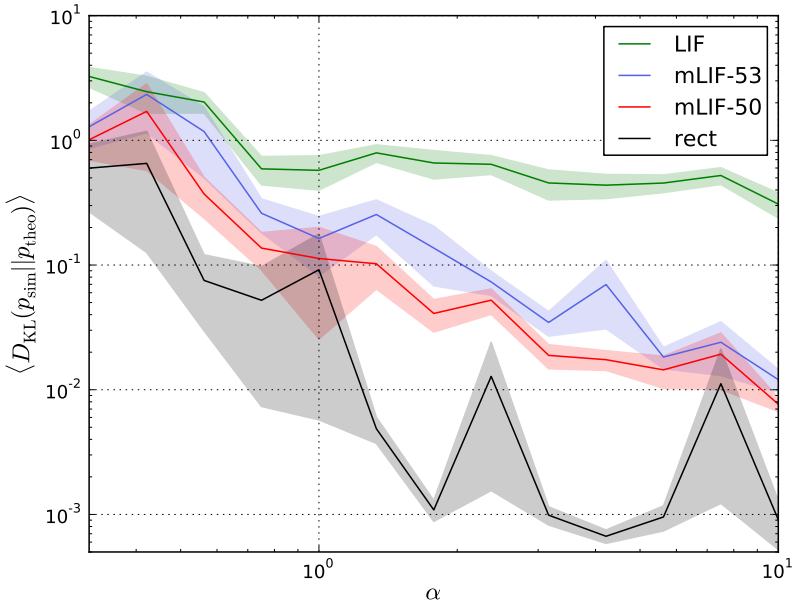


Figure 4.17: Sampling performance of LIF neurons and stochastic neurons on general Bayesian networks as a function of the Dirichlet parameter α (see Figure 4.16). Both mLIF PSP implementations show a better sampling performance of up to one order of magnitude, in terms of KL divergence between the sampling and the target probability distribution, compared to LIF PSPs. The shaded regions denote the standard error of the mean.

4.6 Conclusion

This chapter showed that it is possible to transfer Implementation 1, or in general words, sampling from probability distributions described by BNs to the dynamics of LIF neurons in a spiking noisy environment.
A sophisticated modification of the LIF PSPs led to a vast improvement of the LIF sampling performance from general BNs. The so-called mLIF PSPs were designed by an extension of the connectivity pattern of Implementation 1 by chains of parrot-like neurons. In difference to the LIF PSPs, the new PSPs were close to the ideal rectangular PSPs and had a finite duration. The introduced mLIF PSPs allowed for a sampling performance from general BNs which was up to one order of magnitude more precise than sampling with LIF PSPs in terms of KL divergence between the simulated and the target joint probability distribution.
A disfavor of the mLIF PSPs was the slow convergence in the range of 100 s and longer which arises due to the introduced auxiliary RVs. However, the same slow convergence of Implementation 1 as for mLIF PSPs was observed for the abstract neuron model with rectangular and alpha-shaped PSPs. Inherent LIF PSPs, indeed, allowed for a convergence

time in the range of 10 s. This can be explained by the asymmetry of the PSP shape and, thus, a fast settling towards some stationary distribution.
With regard to a possible application of Implementation 1 on the BrainScaleS neuromorphic hardware system (see Section 2.2), the mLIF PSPs were tested for an optimal and a hardware-compatible distance between the threshold and reset voltage of the sampling neurons. The sampling performance of the hardware-compatible version was close to the optimal one and more precise than the sampling quality with standard LIF PSPs. What will be more crucial for an application on hardware are for both LIF PSPs and mLIF PSPs the time constants and delays (see Chapter 6).

5 Bayesian Networks: Implementation 2 with LIF Neurons

The simulation results in Chapter 4 demonstrated that Implementation 1 (see Section 2.1.5) allows for sampling from BNs via networks of LIF neurons. It was shown for general BNs that so-called mLIF PSPs, which aim at mimicking the ideal rectangular PSP shape of the abstract neuron model, yield better sampling results than the original LIF PSPs. However, one disadvantage of Implementation 1 in general, which results due to the increased number of RVs and, consequently, a higher number of computational units, was the slow convergence in the range of 100 s towards a stationary distribution.
Pecevski et al. [2011] introduce a further method which uses auxiliary neurons to implement neural sampling from BNs. In opposite to the method described in Chapter 4, in this so-called Implementation 2 (see Section 2.1.5) the neurons which correspond to the original RVs directly satisfy the NCC (see Equation 2.18), and not only in the extended probability distribution $p(\mathbf{z}, \mathbf{x})$ with auxiliary RVs $\mathbf{X}$. According to *Pecevski et al.* [2011], this circumstance allows for a much faster convergence to the stationary distribution.
Implementation 2 uses a Markov blanket expansion of the NCC to provide the ability to sample from BN. For each Markov blanket B_k in the graphical model, which is the set of all parents, co-parents and children of the node k, auxiliary neurons $\alpha_k^{\mathbf{v}}$ are introduced, which code for all possible states $\mathbf{v}$ of the Markov blanket. An auxiliary neuron $\alpha_k^{\mathbf{v}}$ is only allowed to spike, if the Markov blanket occupies its corresponding state $\mathbf{v}$. In this case, the membrane potential of the auxiliary neuron $\alpha_k^{\mathbf{v}}$ satisfies the NCC. Each Z_k of the actual RVs $\mathbf{Z}$ of the probability distribution $p(\mathbf{z})$ is represented via a *principal* neuron ν_k. The firing of an auxiliary neuron $\alpha_k^{\mathbf{v}}$ triggers an immediate firing of the corresponding principal neuron ν_k which ensures that also the principal neurons satisfy the NCC.
Section 5.1 introduces the concrete implementations of two BNs which were used to test the sampling performance with LIF neurons. The first BN is a network which consists of two RVs whose joint probability distribution is a uniform distribution. The second BN is the probabilistic model of the Visual Perception Experiment (see Section 2.1.3).
Section 5.2 presents the sampling performance of LIF neurons for the two described BNs when using Implementation 2. The LIF sampling results are compared with these of stochastic neurons with rectangular or alpha-shaped PSPs. It is shown that the non-uniformity of the LIF PSP shape facilitates a correlated spiking activity among the principal neurons and, thus, is responsible for the unsatisfactory sampling results of Implementation 2.
Section 5.3 entails the sampling results of LIF neurons with mLIF PSPs, which were introduced in Section 4.4. Neurons incorporating mLIF PSPs show a better sampling performance compared to those with standard LIF PSPs. However, the sampling quality

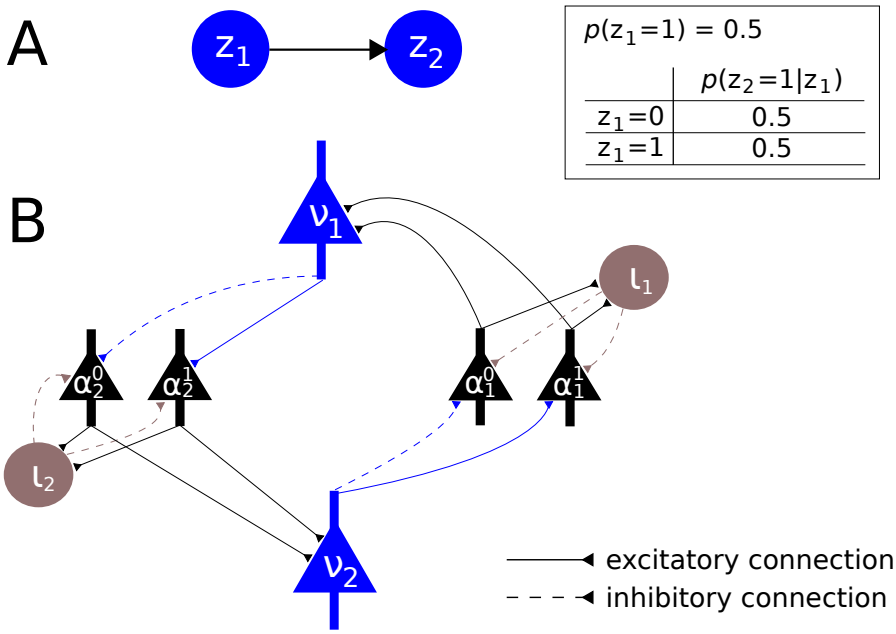


Figure 5.1: Demonstration of Implementation 2 (see Section 2.1.5) for the so-called Two-Node-Network (TNN), a Bayesian network of two random variables whose joint probability distribution follows a uniform distribution. **A)** Graphical model and prior probabilities. **B)** Its neural representation. Each of the auxiliary neurons (*black*) satisfies the NCC (Equation 2.18). The principal neurons (*blue*) and the inhibitory interneurons (*gray*) fire on each incoming spike.

is unsatisfactory compared to the results of Implementation 1 in Section 4.4. A vast improvement of the sampling quality can only be achieved with distinctly longer chains of neurons, which goes at the expense of the overall network size and, thus, the duration of simulations.
Finally, Section 5.4 provides for a theoretical explanation of the dissatisfying sampling performance with mLIF PSPs based on the distributions of first passage times. It is shown that substantial imprecisions arise due to the non-uniformity of the firing probability, the stochasticity induced by the OU process of LIF neurons in a noisy environment in contrast to the Bernoulli process of abstract neurons, and the autocorrelation of the membrane potential.

5.1 Implementation 2: Illustrative Implementation of Sample Bayesian Networks

This section presents the two BNs which are used as example probability distributions for testing the sampling performance of LIF neurons with Implementation 2.

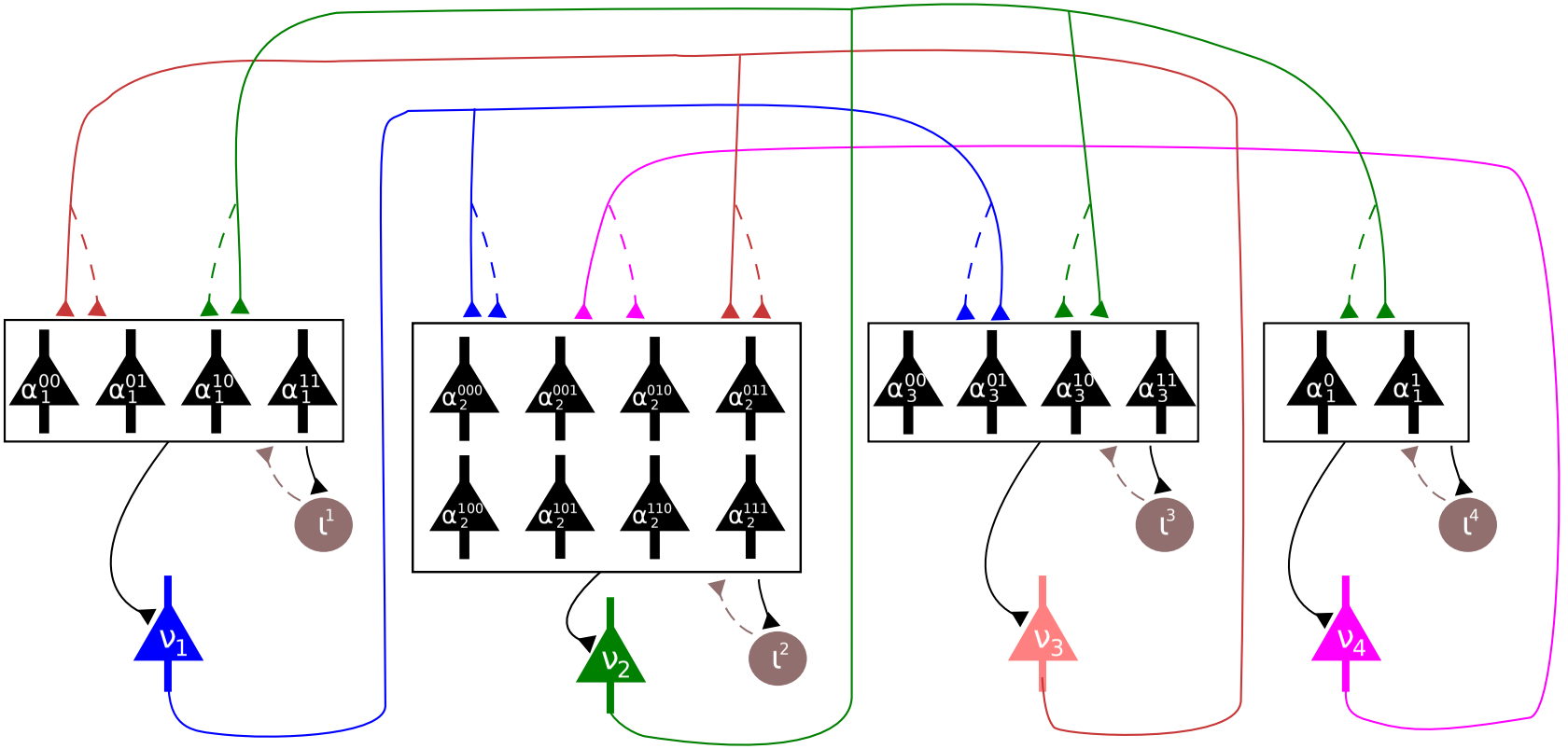


Figure 5.2: Implementation 2 of the VPE. The neural network consists of the four principal neurons $\nu_1, \nu_2, \nu_3, \nu_4$, their corresponding auxiliary neurons (*black*) and inhibitory interneurons (*gray*). For reasons of clarity, not all connections from principal to auxiliary neurons and between the auxiliary neurons and inhibitory interneurons are drawn. Section 2.1.5 contains a detailed description of the implementation of a Markov blanket.

5.1.1 Implementation of the Two-Node-Network

The first BN is a uniform probability distribution of two RVs Z_1 and Z_2 of the shape

$$p\left(z_1, z_2\right) = p\left(z_1\right)\, p\left(z_2|z_1\right)\ . \tag{5.1}$$

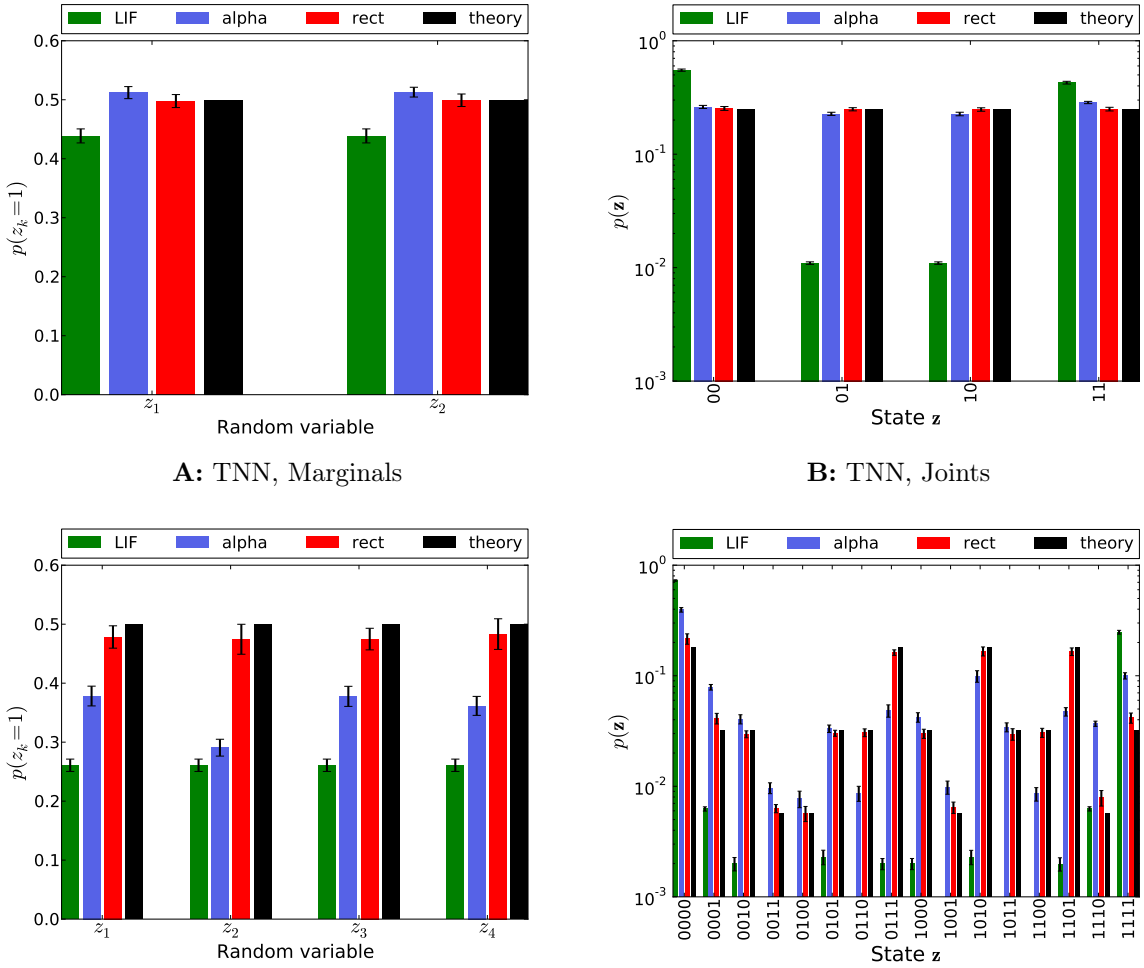
Figure 5.1A shows the graphical model of the BN and the prior and conditional probabilities $p\left(z_1 = 1\right)$ and $p\left(z_2 = 1|z_1\right)$, which completely describe the BN. In the following, the network will be referred to as Two-Node-Network (TNN).
Figure 5.1B shows the neural implementation of the BN in terms of Implementation 2. The task involves 8 neurons: the two principal neurons ν_1 and ν_2 represent the two nodes Z_1 and Z_2. The RV Z_1 constitutes the Markov blanket B_2 of Z_2 and Z_2 constitutes the Markov blanket B_1 of Z_1. Therefore, each Markov blanket can have two possible states, which are coded by the auxiliary neurons α_k^0 and α_k^1, where ν_k is their corresponding RV. Furthermore, each group of auxiliary neurons is connected to an inhibitory interneuron ι which ensures that after one of the auxiliary neurons α_k^0 and α_k^1 has emitted a spike, both of them remain silent during the refractory period.
Since the membrane potential of the auxiliary neurons satisfies the NCC, the biases of the auxiliary neurons are calculated via Equation 2.31. In the case of the TNN, both α_1^0 and α_2^0 have the bias 0 and α_1^1 and α_2^1 occupy the bias -10 in the stochastic neuron model. Equation 2.32 is used to calculate the excitatory weights from the principal neurons to the auxiliary neurons. The excitatory weights of the connections $\nu_1 \to \alpha_2^1$ and $\nu_2 \to \alpha_1^1$ amount to 10. The inhibitory weights of the connections $\nu_1 \to \alpha_2^0$, $\nu_2 \to \alpha_1^0$, as well as the weights from the inhibitory interneurons ι_k to the auxiliary neurons are calculated via Equation 2.33. Each of these weights amount to -10 in the stochastic neuron model. The strengths of all outgoing connections of the auxiliary neurons are chosen such that a spike of an auxiliary neuron directly evokes a spike of the postsynaptic neuron. For the stochastic neurons the weight is 30 and for LIF neurons with the parameters from Table A.3 the weight amounts to $0.16\,\mu$S.

5.1.2 Implementation of the Visual Perception Experiment

The second BN is the probability distribution which models the VPE described in Section 2.1.3. It is described by the probability distribution

$$p(z_1, z_2, z_3, z_4) = p(z_1)\, p(z_2)\, p(z_3|z_1, z_2)\, p(z_4|z_2)\ . \tag{5.2}$$

Figure 5.2 shows the neural implementation of the VPE. The Markov blankets of the RVs Z_1 to Z_4 are: $B_1 = (Z_2, Z_3)$, $B_2 = (Z_1, Z_3, Z_4)$, $B_3 = (Z_1, Z_2)$ and $B_4 = (Z_2)$. The neural implementation therefore involves 26 neurons: 4 principal neurons, 4 inhibitory interneurons, 4 auxiliary neurons of z_1, 8 auxiliary neurons of z_2, 4 auxiliary neurons of z_3 and 2 auxiliary neurons of z_4. The concrete implementation of biases and weights is accomplished via the Equations 2.31, 2.32 and 2.33 using the prior and conditionals probabilities introduced in Figure 4.1.



A: TNN, Marginals

B: TNN, Joints

C: VPE, Marginals

D: VPE, Joints

Figure 5.3: Comparison of the sampling performance of Implementation 2 with LIF PSPs with parameters from Table A.2, alpha-shaped and rectangular PSPs from the stochastic neuron model, and the target probabilities for the TNN (*upper row*) and the VPE (*lower row*). The left column shows the marginal distributions and the right column the corresponding joint probability distributions. The bars show the average results of 20 simulations of duration 20 s. The error bars denote the standard deviations.

5.2 Implementation 2: Performance on Sample Bayesian Networks

In the following, the sampling results of a LIF-based version of Implementation 2 are presented and compared to the sampling results of stochastic neurons with rectangular and alpha-shaped PSPs. The example probability distributions are the two distributions TNN and VPE which were described in Section 5.1.

5.2.1 Results of Sampling from the Two-Node-Network and the Visual Perception Experiment

Table A.2 lists the parameters of the auxiliary neurons. The principal neurons and inhibitory interneurons adopt the parameter set of Table A.3. They are designed such that upon each incoming spike they elicit exactly one spike. The principal neurons and inhibitory interneurons do not receive background stimuli and the connections to these neurons are static.
For both probability distributions, 20 simulations of 20 s are executed and the average results of sampling from the joint probability distribution of the involved principal neurons are measured. The small standard deviations of the sampling results justify the short duration 20 s of one simulation compared to the ones obtained via Implementation 1.
Figure 5.3 illustrates the sampling results: Panels A and B presents the results of sampling from the TNN while Panels C and D show the results of sampling from the VPE. In comparison to the sampling performance of stochastic neurons with rectangular or alpha-shaped PSPs, the LIF neurons exhibit substantial inaccuracies when sampling from both probability distributions. The most salient characteristic of the sampling results of LIF neurons is the U-like shape of the joint probability distributions. This is caused by the correlated spiking activity of auxiliary neurons, which leads to a vast overrepresentation of the states $(z_1 = 0, z_2 = 0)$ and $(z_1 = 1, z_2 = 1)$ for the TNN and $(z_1 = 0, z_2 = 0, z_3 = 0, z_4 = 0)$ and $(z_1 = 1, z_2 = 1, z_3 = 1, z_4 = 1)$ for the VPE, indicating a significant tendency to synchronize the dynamics.

5.2.2 Theoretical Explanation of the Insufficiency of LIF Sampling when Applying Implementation 2

An explanation for the insufficiency of of LIF Sampling when applying Implementation 2 is sketched in Figure 5.4. It shows a simplified sketch of the membrane potential traces without the Gaussian jitter due to the Poisson distributed background input. The starting point is the network state $(z_1 = 0, z_2 = 0)$. The illustration begins in Panel A with a spike of the auxiliary neuron α_1^0 according to the NCC at time t_1. This action potential triggers a firing of the inhibitory interneuron ι_1 which leads to an inhibitory PSP in the membrane voltage trace of α_1^0. Furthermore, the principal neuron ν_1 is triggered to elicit an immediate spike at time $t_1 + \Delta t$ (Panel B), which changes the network state to $(z_1 = 1, z_2 = 0)$. The expression Δt hereby denotes the combination of the interneural delay and the rise time of the membrane potential.

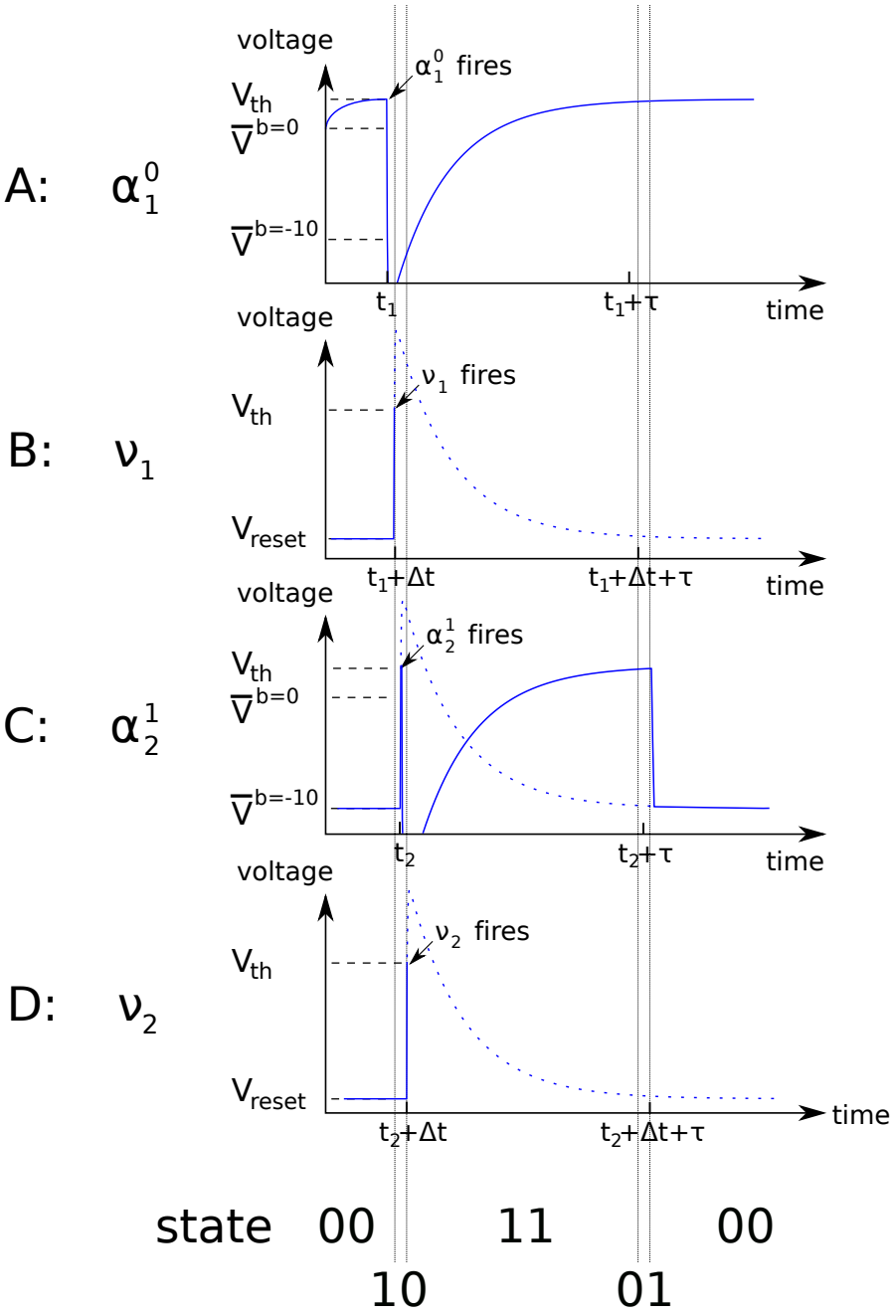


Figure 5.4: A simplified sketch of the mean membrane voltage traces for the neurons from the TNN, which shows the impact of the asymmetry of the LIF PSP shape on the sampling distribution of network states. After the neuron α_1^0 has elicited a spike **(A)**, it forces ν_1 to spike directly **(B)** which itself raises the membrane potential of neuron α_2^1 to the average potential $\overline{V}^{b=0}$. The firing probability during the first milliseconds of the LIF PSP is much higher than the firing probability during the residual part. Thus, α_2^1 is very likely to spike very soon after it has been activated **(C)**. The firing of α_2^1 directly triggers ν_2 to elicit a spike **(D)**. This leads to the overrepresentation of the states $(z_1 = 0, z_2 = 0)$ and $(z_1 = 1, z_2 = 1)$ in the sampling probability distribution decreasing the frequency of the states $(z_1 = 0, z_2 = 1)$ and $(z_1 = 1, z_2 = 0)$ (see Figure 5.3).

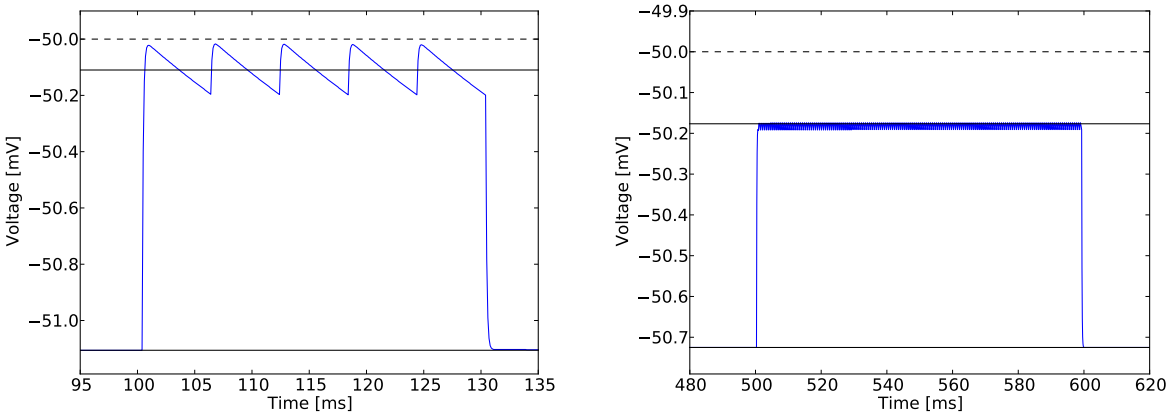
The firing of the principal neuron ν_1 decreases the average membrane potential of α_2^0 to $\overline{V}^{b=-10}$ and, at once, raises the average membrane potential of α_2^1 to $\overline{V}^{b=0}$. Due to the asymmetric PSP shape the α_2^1 spikes with a very high probability at the beginning of the refractory period and a very low probability close to τ_{ref}. This discrepancy, compared to the stochasticity of the abstract neuron model, favors a soon spiking of α_2^1 at time t_2 as shown in Panel C. The spike of α_2^1 immediately triggers ν_2 to spike at $t_2 + \Delta t$ (Panel D), which changes the network state to $(z_1 = 1, z_2 = 1)$. In average, this quick change of states is the reason that the states $(z_1 = 0, z_2 = 1)$ and $(z_1 = 1, z_2 = 0)$ are only assumed for a short fraction of time, which results in the U-like shape of the joint probability distribution in Figure 5.3B.
A further error source is the too large residual of a LIF PSP on the course of the membrane potential after τ_{ref} (see Figure 4.7). Since the overall network activity has to be spontaneously initiated by the firing of one of the α^0 neurons, a prolonged inhibition due to this effect explains the higher frequency of the states $(z_1 = 0, z_2 = 0)$ as compared to $(z_1 = 1, z_2 = 1)$ in the sampling result of LIF neurons in Figure 5.3B.
The large deviations of the simulated joint probability distribution of the VPE from the theoretical distribution can be reduced to the same drawbacks. The overrepresentation of the states $(z_1 = 0, z_2 = 0, z_3 = 0, z_4 = 0)$ and $(z_1 = 1, z_2 = 1, z_3 = 1, z_4 = 1)$ in Figure 5.3D at the reduces probability of occurrence of the remaining states follows directly from the asymmetry and (endless) exponential decay of the PSP shape, which were described in Figure 5.4.

5.3 Improvement of LIF Sampling via mLIF PSPs

The unsatisfactory LIF sampling results from Section 5.2 raise the question whether it is possible to improve the sampling performance with mLIF PSPs, a method which was already successfully applied for Implementation 1 in Section 4.4. A PSP shape which is closer to the ideal rectangular shape would have two advantages. Firstly, the symmetric shape would not favor the firing at a special point in time during the refractory period. Secondly, due to the instant decay of the PSP at the end of the refractory period, the subsequent course of the membrane potential, and therefore the spiking probability, will not be influenced beyond the refractory period.

5.3.1 Engineering mLIF PSPs

Figure 5.5 shows the two mLIF PSP shapes which were used in this section. The PSP shape in Panel A is generated with chains of 6 neurons and thus will be referred to as mLIF_6 PSP. Table A.4 contains the parameters that are necessary to design the mLIF_6 PSP, with the reset voltage of the sampling neurons $V_{\text{reset}} = -50.01\,\text{mV}$. The mLIF_6 PSP already provided for sufficiently precise sampling results with Implementation 1. Panel B shows the so-called mLIF_{166} PSP. It is generated via chains of 166 neurons. The utilized parameters are listed in Table A.5.
Only the principal neurons and the inhibitory interneurons are replaced by neural chains,



A: mLIF_6 **B:** mLIF_{166}

Figure 5.5: The two mLIF PSP shapes which are used to investigate the feasibility of Implementation 2 with LIF neurons. Each of the PSP shapes is constructed by a chain of multiple principle neurons realizing the method from Figure 4.11B. In each of the voltage plots the *dashed* line represents the threshold potential, the *upper solid* line is $\overline{V}^{b=0}$ (no bias) and the *lower solid* line is $\overline{V}^{b=-10}$, the potential which corresponds to the bias -10. Both figures show a PSP shape induced by an incoming spike via a connection of strength 10.

which is in contrast to Implementation 1. The principal neurons and the inhibitory interneurons indeed require the asymmetric shape of a standard LIF PSP, such that with the parameters listed in Table A.4 and A.5, they are triggered to elicit a spike upon each incoming spike from their corresponding auxiliary neurons. Therefore, the spikes of the auxiliary neurons do not need to be forwarded by parrot neurons.
In contrast to mLIF_6 PSPs, two consecutive mLIF_{166} PSP are completely uncorrelated which can be deduced from Figure 5.6. The synaptic time constant τ_{syn} hereby can be associated with a "memory" of the LIF neuron in the HCS. For $\tau_{\text{syn}} = 30\,\text{ms}$, which corresponds to an mLIF_6 PSP, the autocorrelation of the membrane voltage is much larger than 0 at $\tau_{\text{ref}} = 30\,\text{ms}$. For $\tau_{\text{syn}} = 10\,\text{ms}$, which indeed corresponds to an mLIF_{166} PSP, the "memory" of the LIF unit is shorter, such that the autocorrelation function is approximately 0 at $\tau_{\text{ref}} = 100\,\text{ms}$, which is exactly like the autocorrelation of the membrane potential in the stochastic neuron model.

5.3.2 Results of LIF Sampling with mLIF PSPs

The sampling performance of LIF neurons with LIF PSPs, mLIF_6 PSPs and mLIF_{166} PSPs is measured for the two probability distributions which are introduced in Section 5.1. The total number of neurons which is needed to sample from the TNN amounts to 28 neurons for mLIF_6 PSPs and 668 neurons for mLIF_{166} PSPs. The number of neurons

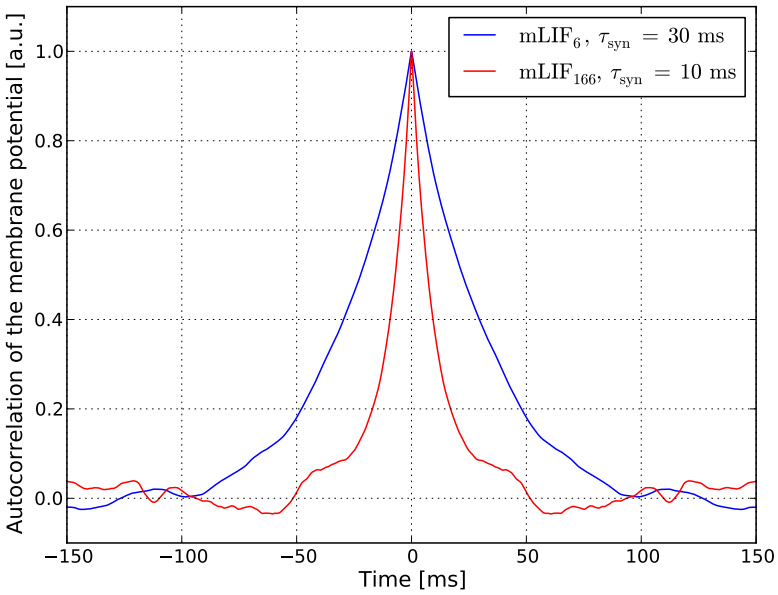
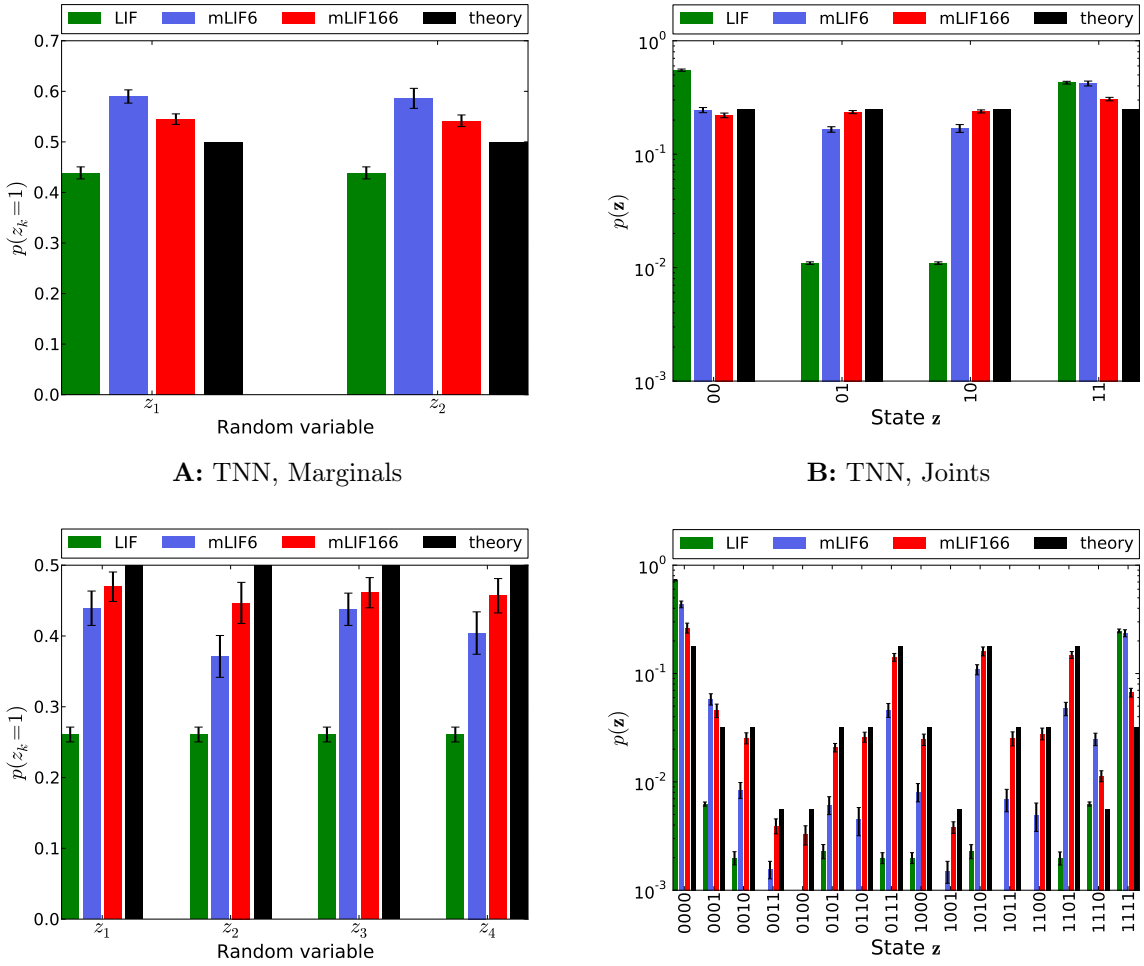


Figure 5.6: The relative autocorrelation function of the membrane potential of a LIF neuron with $\tau_{\text{syn}} = 30\,\text{ms}$, which refers to mLIF$_6$ PSPs, or $\tau_{\text{syn}} = 10\,\text{ms}$, which refers to mLIF$_{166}$ PSPs, with respect to the maximal autocorrelation. For mLIF$_6$ PSPs, autocorrelations of the membrane potential exist even after the refractory period $\tau_{\text{ref}} = 30\,\text{ms}$. In contrast, the autocorrelation of the membrane potential for mLIF$_{166}$ PSPs are about 0 at $\tau_{\text{ref}} = 100\,\text{ms}$.

which is indeed needed to sample from the VPE network is 66 neurons for mLIF$_6$ PSPs and 1346 neurons for sampling with mLIF$_{166}$ PSPs.
Figure 5.7 presents the sampling results of 20 simulations of duration 20 s for LIF PSPs, 30 s for mLIF$_6$ PSPs and 100 s for mLIF$_{166}$ PSPs. The different simulation times are chosen due to differing refractory times τ_{ref}, such that in all three cases the neurons have the same number of opportunities to change their state.
Both implementations with mLIF PSPs outperform the sampling results with LIF PSPs. The results with mLIF$_{166}$ PSPs, however, are much closer to the target probabilities. mLIF$_6$ PSPs show a less precise sampling performance than stochastic neurons with alpha-shaped PSPs in Figure 5.3. Similar to the results with LIF PSPs in Figure 5.3, mLIF$_6$ PSPs lead to a U-like shape of the sampled joint probability distribution.
This observation needs to be discussed in the following, since the mLIF PSP shapes were primarily engineered in order to resolve the drawbacks due to the asymmetry of the LIF PSP shape and its influence on the membrane potential beyond the refractory period. The insufficient results of mLIF$_6$ PSPs, on the one hand, and the satisfactory results for mLIF$_{166}$ PSPs, on the other hand, however, show that these can not be the only factors which influence the sampling ability when using Implementation 2. The next section



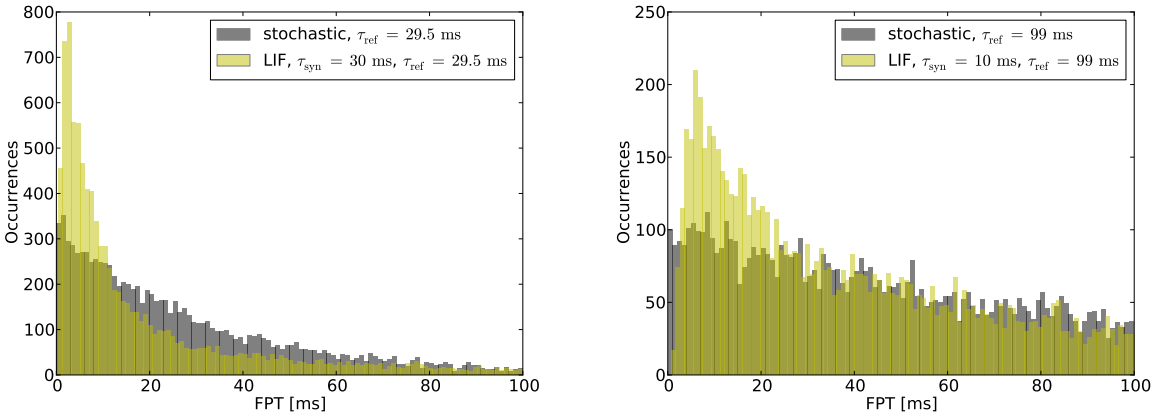
A: TNN, Marginals

B: TNN, Joints

C: VPE, Marginals

D: VPE, Joints

Figure 5.7: Comparison of the sampling performance for simulations of Implementation 2 with LIF PSPs with parameters from Table A.2, mLIF_6 PSPs, mLIF_{166} PSPs and the target probabilities for the TNN (Panels **A** and **B**) and the VPE (Panels **C** and **D**). The left column shows the marginal distributions and the right column the corresponding joint probability distributions. The bars show the average results of 20 simulations of duration 20 s for LIF PSPs, 30 s for mLIF_6 PSPs and 100 s for mLIF_{166} PSPs. The error bars denote the standard deviations.



A: FPT distribution for $\tau_{\text{ref}} = 29.5\,\text{ms}$ **B:** FPT distribution for $\tau_{\text{ref}} = 99\,\text{ms}$

Figure 5.8: Comparison of the FPT distributions of stochastic neurons and LIF neurons with $\tau_{\text{ref}} = 29.5\,\text{ms}$ **(A)**, which is the duration of the mLIF_6 PSP shape, and $\tau_{\text{ref}} = 99\,\text{ms}$ **(B)**, which is the duration of the mLIF_{166} PSP shape, for the auxiliary neuron α^0 from the TNN. The stochastic neuron model allows for an instant spiking. Thus, the FPT distribution of the stochastic neuron model is exponential, while the FPT distribution of LIF neurons is Poisson-shaped. Both distributions of first passage times of LIF neurons are characteristic for an OU process (see e.g. *Gotoh et al.* [2011]).

deals with the investigation of the distribution of first passage times for both, mLIF_6 and mLIF_{166} PSPs, and reveals that the distribution of spike times is crucial for a satisfactory sampling performance via Implementation 2.

5.4 Investigation of the Distributions of First Passage Times of LIF Neurons

Section 5.2 concluded based on simulation results that the LIF sampling performance with standard LIF PSPs using Implementation 2 is unsatisfactory due to the asymmetric PSP shape and the exponential decay of the PSP shape and, thus, an influence on the course of the membrane potential beyond the refractory period.
With the help of mLIF PSPs, we eliminated both error sources (see Section 5.3), which distinctly improved the sampling results. However, compared to the sampling results of Implementation 1 in Section 4.4, these results are still unsatisfactory. In order to investigate the inferior sampling quality of LIF neurons with mLIF_6 PSPs, a detailed analysis of the first passage times of the auxiliary neurons α^0 and α^1 of the TNN in Figure 5.1 is performed in the following.

5.4.1 Distribution of First Passage Times for α^0 from the Two-Node-Network

Figure 5.8 shows the distributions of first passage times of stochastic neurons and LIF neurons for the auxiliary neuron α^0 from the TNN. Panel A shows the distributions for $\tau_{\text{ref}} = 29.5\,\text{ms}$ and $\tau_{\text{syn}} = 30\,\text{ms}$ which are the parameters of the mLIF_6 PSP shape, while Panel B illustrates the distributions for $\tau_{\text{ref}} = 99\,\text{ms}$ and $\tau_{\text{syn}} = 10\,\text{ms}$, which refer to the mLIF_{166} PSP shape. For the stochastic neuron, we have $\tau_{\text{ref}} = \tau_{\text{syn}}$. The histograms are results of 10000 simulations of 100 ms each. For each run, the bias of the neurons is initialized to 0. It is a model e.g. for the auxiliary neuron α_1^0 of Figure 5.1 if it is not inhibited by the interneuron ι_1 or the active principal neuron ν_2.
The distribution of the first passage times has an exponential shape for stochastic neurons and a Poisson shape for LIF neurons, which is due to τ_{m}, which is 0 for stochastic neurons and larger than 0 for LIF neurons. The histograms in Figure 5.8 indicate an increased firing probability of LIF neurons at the beginning of the refractory period compared to the firing probability of the stochastic neuron model. These dynamics favor a synchronous spiking activity of the neurons which was also observed for neurons with LIF PSP in Section 5.2.
The distribution for LIF neurons in Panel B approximates more accurately to the distribution of the stochastic neuron model compared to the distribution in Panel A. This partly explains the better sampling performance using mLIF_{166} PSPs compared to the performance of mLIF_6 PSPs in Section 5.3.
The following subsection investigates the distribution of FPTs for the neuron α^1 from the Two-Node-Network and entails a further aspect which leads to the unsatisfactory sampling performance in Section 5.3.

5.4.2 Distribution of First Passage Times for α^1 from the Two-Node-Network

Figure 5.9 illustrates an experiment which comprises a further explanation of the unsatisfactory sampling performance via mLIF_6 PSPs. The setup measures the distribution of first passage times for the auxiliary neurons α^1 from the TNN in Figure 5.1 if they are not inhibited by the interneurons. Without an external stimulus, the neuron α^1 has an average membrane potential $\overline{V}^{b=-10}$. Due to the Poisson background stimulus the membrane potential exhibits stochastic fluctuations around this average membrane potential. If the principal neuron ν_1^1 elicits a spike, the mean membrane potential of α_2^1 is raised to $\overline{V}^{b=0}$ for the duration of the refractory period. Now the first passage time $\Delta t\left(\nu_1^1, \alpha_2^1\right)$ is measured, after which the auxiliary neuron fires for the first time.

Evaluation of the FPT Distribution of α^1 Figure 5.10 shows the distribution of first passage times of the auxiliary neuron α_2^1 which is measured in 10000 simulation runs using the method described in Figure 5.9. Panel A and B show the distributions of first passage times for the two respective mLIF PSP shapes in Figure 5.5. The main characteristic of

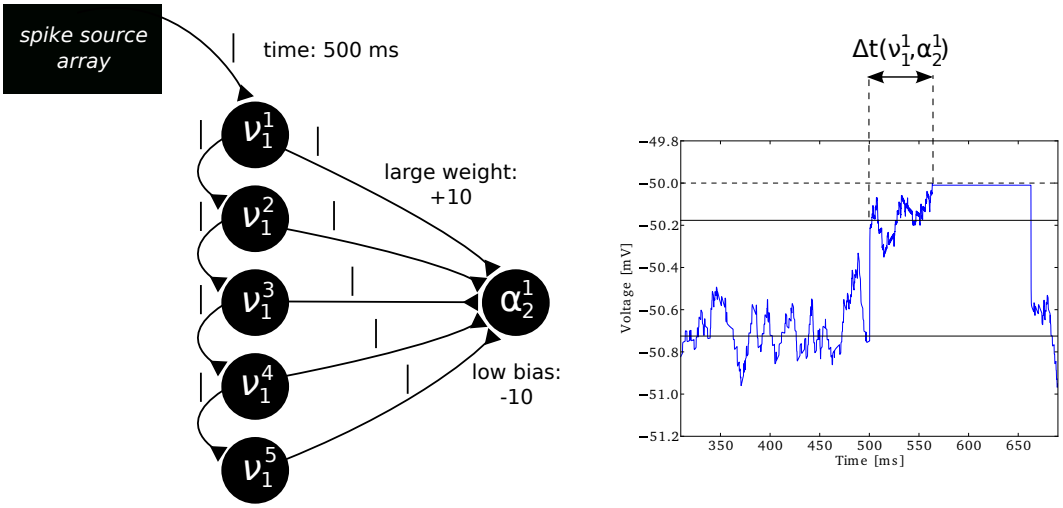
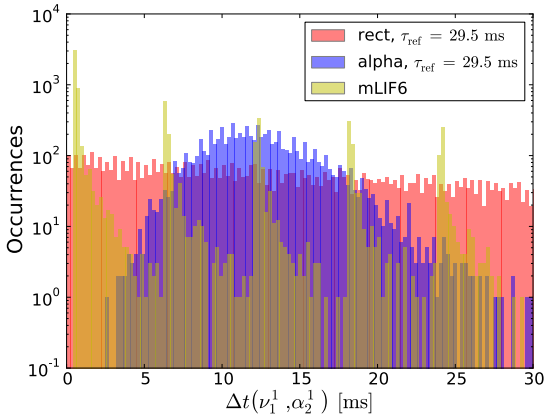
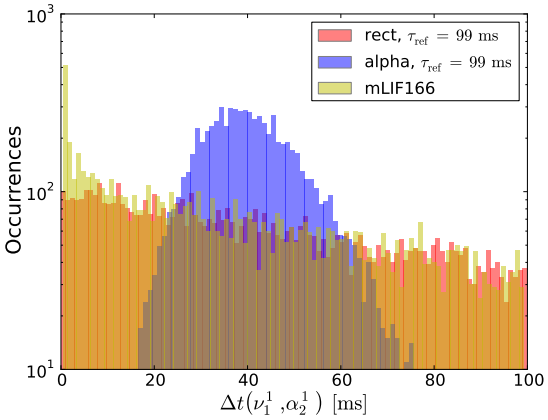


Figure 5.9: Feasibility test of Implementation 2 with LIF neurons and PSP shapes from Figure 5.5: The auxiliary neuron α_2^1 has a mean membrane potential $\overline{V}^{b=-10}$, which corresponds to the bias -10 in the stochastic neuron model. At the point 500 ms, a spike of an external spike source triggers the firing of the chain of K principal neurons $\nu_1^1, ..., \nu_1^K$ which provide an mLIF PSP shape in the voltage trace of α_2^1. The weight from the chain of principal neurons to the auxiliary neuron corresponds to the weight 10 in the stochastic neuron model. The first passage time $\Delta t\left(\nu_1^1, \alpha_2^1\right)$ is measured, which is the time after which α_2^1 emits a spike once its mean membrane potential has been raised to $\overline{V}^{b=0}$ by the input of the principal neurons.

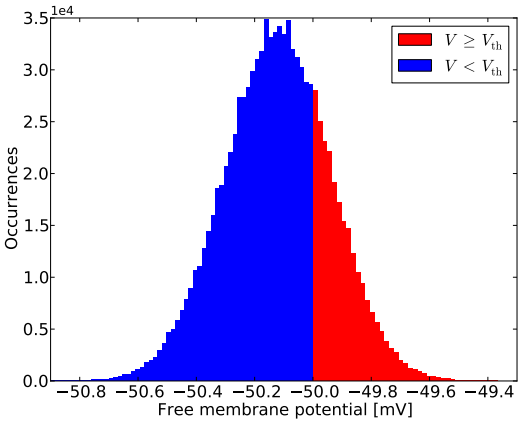
both distributions of LIF neurons are the large peaks within the first few milliseconds compared to distributions of stochastic neurons with rectangular or alpha-shaped PSPs. In the FPT distribution of the mLIF_6 PSP in Figure 5.10A 5 peaks can be distinguished which coincide with the time positions of the edges of the "teeth" of the corresponding PSP shape in Figure 5.5. The first peak is more than an order of magnitude higher than the largest value of the exponentially-shaped distribution of the stochastic neuron with a rectangular PSP shape. This means: if the mean membrane potential of the neurons α_1^1 and α_2^1 is increased to $\overline{V}^{b=0}$ by an incoming spike from a principal neuron, they either emit a spike immediately or, otherwise, tend to remain silent for the whole refractory period of the principal neurons. In addition to the FPT distribution of α_1^0 and α_2^0 in Figure 5.8, this fact is a further and even more intuitive explanation for the unsatisfactory sampling performance of LIF neurons with mLIF_6 PSPs in Figure 5.7.
In contrast to the distribution of the mLIF_6 PSP, the FPT distribution when using mLIF_{166} PSP in Panel A fits well to the distribution of rectangular PSPs for the most part of the interval. This explains the clearly superior sampling performance of mLIF_{166} PSPs in Figure 5.7. The large first peak explains the small overrepresentation of the states $(z_1 = 1, z_2 = 1)$ in Figure 5.7B: the neurons α_1^1 and α_2^1 spike consecutively slightly



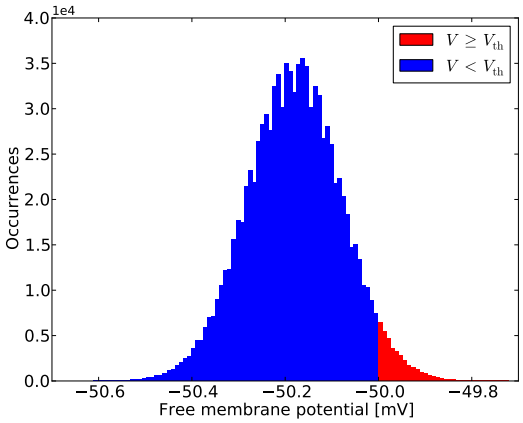
A: Distribution of $\Delta t\left(\nu_1^1, \alpha_2^1\right)$ for $\tau_{\text{ref}} = 29.5\,\text{ms}$



B: Distribution of $\Delta t\left(\nu_1^1, \alpha_2^1\right)$ for $\tau_{\text{ref}} = 99\,\text{ms}$

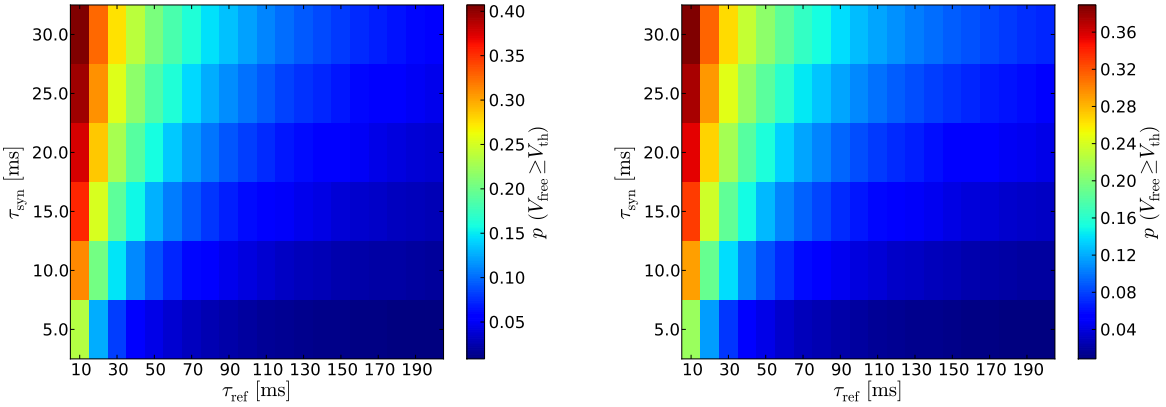


C: Distribution of V_{free} for a LIF neuron with $\tau_{\text{ref}} = 29.5\,\text{ms}$



D: Distribution of V_{free} for a LIF neuron with $\tau_{\text{ref}} = 99\,\text{ms}$

Figure 5.10: Panels **A** and **B** show the unnormalized probability densities that a firing after the time Δt occurs once the auxiliary neuron α^1 for the TNN has been activated from voltage $\overline{V}^{b=-10}$ to $\overline{V}^{b=0}$ (see Figure 5.9). The distributions for rectangular PSPs, alpha-shaped PSPs and mLIF PSPs are illustrated with $\tau_{\text{ref}} = 29.5\,\text{ms}$ (*left*) and $\tau_{\text{ref}} = 99\,\text{ms}$ (*right*). Panels **C** and **D** show the distributions of the free membrane potential of the auxiliary neuron α_1^1, which is elevated to the average membrane potential $\overline{V}^{b=0}$ by stringing together consecutive mLIF PSP. This mimics the input of ν_2 with z_2 clamped to 1. The *blue* part of the distributions is below the threshold voltage while the *red* part is above.



A: Background rate $\nu = 400\,\text{Hz}$ and weight $w = 2\,\text{nS}$

B: Background rate $\nu = 6400\,\text{Hz}$ and weight $w = 0.5\,\text{nS}$

Figure 5.11: Color-coded probability $p\,(V_{\text{free}} \geq V_{\text{th}} = -50\,\text{mV})$ of the free membrane potential of a LIF neuron, whose mean membrane potential is $\overline{V}^{b=0}$ and which receives a balanced random background input. With larger τ_{ref} and lower τ_{syn} (e.g. mLIF_{166} compared to mLIF_6), the fraction of V_{free} above V_{th} becomes smaller and, therefore, the sampling performance will improve (see Figure 5.10). The simulations are run with different background rates and weights, both of which are chosen such that the standard deviation of the free membrane potential is equal, but the average free membrane potential is lower in Panel **B** compared to Panel **A** (see Equation 2.54). However, the shift of the average membrane potential to lower values is too small to expect a noticeable improvement of the sampling quality.

more frequently than the stochastic neurons with rectangular PSPs and, thus, decrease the proportion of the states $(z_1 = 0, z_2 = 1)$ and $(z_1 = 1, z_2 = 0)$.

Evaluation of the Distribution of the Free Membrane Potential of α^1 The large first peaks of the distributions of first passage times in Figure 5.10 can be explained by evaluating the distribution of the free membrane potential of a LIF neuron with average membrane potential $\overline{V}^{b=0}$ which is plotted in Figure 5.10C and D for the respective PSP shapes in Figure 5.5. For both distributions, the red bars denote the fraction of the membrane potential above the threshold voltage and the blue bars denote the fraction of the membrane potential below the threshold voltage.
The fluctuations of the membrane voltage around some mean membrane potential induced by the Poisson-distributed background input are equal for the mean membrane potentials $\overline{V}^{b=-10}$ and for $\overline{V}^{b=0}$. This implies that the probability of the neurons α_1^1 and α_2^1 to fire immediately after the increase of the mean membrane potential from $\overline{V}^{b=-10}$ to $\overline{V}^{b=0}$ is proportional to the red area of the corresponding distribution of the free membrane

potential in Figure 5.10C and D. For $\tau_{\text{ref}} = 29.5\,\text{ms}$ and $\tau_{\text{syn}} = 30\,\text{ms}$, which are the time scales of the mLIF_6 PSP, the red area in Figure 5.10C amounts to about 27.2%. This value is in approximate accordance with the relative height of the first peak of the FPT distribution of α^1 in Figure 5.10A. For $\tau_{\text{ref}} = 99\,\text{ms}$ and $\tau_{\text{syn}} = 10\,\text{ms}$, which refers to the mLIF_{166} PSP, the red fraction in Figure 5.10D is about 3.7%, which is also in approximate accordance with the overshoot of the first large peak in Figure 5.10B.
Figure 5.11A illustrates the fraction of the free membrane potential which is larger than the threshold voltage V_{th} for different choices of τ_{syn} and τ_{ref}. In order to achieve a satisfactory sampling result with Implementation 2, it is essential to choose a large τ_{ref} and low τ_{syn}, which means that the autocorrelation of the membrane potential, which bears the "memory" of the LIF neuron, has to vanish at best. However, a large τ_{ref} and low τ_{syn} implies very long chains of neurons which scales exponentially for tasks with an increasing number of RVs and, thus, becomes computationally not feasible.
A question that remains is whether it is possible to decrease the height of the first peak in of the FPT distribution of α^1 in Figure 5.10A by modifying the background input parameters. The new frequency and synaptic input weights can be chosen such that the standard deviation of the membrane potential remains constant but the average membrane potential is shifted to a lower value (see Equation 2.54). A shift of the mean membrane potential to a lower value implies a shrinkage of the red area of Figure 5.10C and consequently a shrinkage of the height of the first peak in of the FPT distribution of α^1 in Figure 5.10A.
Figure 5.11B shows the fraction of the free membrane potential which is larger than the threshold voltage V_{th} for a much larger background input frequency of $\nu = 6400\,\text{Hz}$ and much lower synaptic input weight $w = 0.5\,\text{nS}$ for the excitatory and inhibitory connections. However, this shift of the mean membrane potential turns out to be very small compared to Figure 5.11A and will not significantly improve the sampling results.
The following paragraph comprises two crucial reasons against a modification of the mLIF PSP shape with the aim to improve the sampling quality.

Reasons Against a Further Modification of the mLIF PSP Shape The results of the distribution of first passage times with alpha-shaped PSPs in Figure 5.10 might lead to the assumption that rounder PSP shapes, instead of the imitations of the ideal rectangular PSPs in Figure 5.5, could improve the results of LIF sampling with mLIF_6 PSPs. A rounding of the PSP shape would shift the large peak in Figure 5.10A towards the end of the refractory period of the principal neuron.
However, rendering the first "tooth" of the PSP shape in Figure 5.5 smaller would entail a raising of the remaining "teeth" in order to fulfill the consistency of the integral of individual PSPs of the LIF neuron model and the stochastic neuron model in Equation 2.73. Such an increase would result in a PSP shape crossing the threshold potential by design.
Furthermore, a rounding of the PSP shape would lead to large jumps of the membrane potential e.g. of the auxiliary neuron α_1^1 whenever the RV z_2 is clamped to 1. This results in undesirable threshold crossings (see Figure 4.12B) and distorts the sampling quality.

5.5 Conclusion

This chapter aimed at testing the feasibility of Implementation 2 of BNs with LIF neurons (see Section 2.1.5). Based on simulation results we can conclude that, if the sampling ability depends on the distribution of spike timings, LIF sampling with both original LIF PSPs and mLIF PSPs suffers substantial precision.
This imprecision could be ascribed to three inherent aspects which differ between the LIF neuron model and the abstract neuron model. First, both LIF and mLIF PSPs are non-uniform, which results in a non-uniform firing probability. Second, the stochasticity induced by the OU process, which describes the membrane potential of LIF neurons in a noisy environment, is different from the stochasticity induced by the Bernoulli process, which characterizes the evolution of the membrane potential of abstract neurons. Third, the membrane potential of the LIF neurons is autocorrelated, which is in contrast to the abstract neuron model. Due to this fact, additional correlations of the spiking activity of the sampling neurons were introduced, which deterred from the desired functionality of the networks.
Only for large chains of neurons, yielding large refractory times and low synaptic time constants, adequate sampling results were achieved which are still less promising than the LIF sampling results via Implementation 1 in Chapter 4. In this case, the occurrences of the first threshold crossing were much more distributed in accordance with the distribution of first passage times of the stochastic neuron model.
An application of Implementation 2 with mLIF_{166} PSPs on the BrainScaleS neuromorphic hardware system (see Section 2.2) will be infeasible with the inherent fluctuations of voltages and currents, since mLIF_{166} PSPs require the implementation of chains of 166 neurons which all have the same characteristics.

6 Towards LIF-based Boltzmann Machines on Neuromorphic Hardware

The previous chapters have demonstrated through software simulations that it is possible to physically instantiate Boltzmann Machines and Bayesian Networks with networks of LIF neurons. This chapter investigates the feasibility of performing LIF sampling with the neurons integrated on the HICANN chip described in Section 2.2.1. The demonstrator setup, which is described in Section 2.2.2, provides the necessary communication infrastructure for the chip. The PyHAL API (see Section 2.3.2) is used to control the desired neural and synaptic parameters and run the experiments.

As outlined in Section 2.2, a neuromorphic hardware system entails a broad range of advantages such as defect tolerance, massive parallelism, acceleration compared to biological real-time and low power consumption. However, especially when running experiments with which strongly depend on certain parameter settings, whose precision is crucial for the expected outcome of the experiment, the drawbacks and limitations of the analog elements of the neuromorphic hardware become apparent: the imprecision of floating gate cells, the limitation of parameter ranges, the malfunction of electrical circuits due to production faults or the limitation of the communication bandwidth, to name some examples.

The simulation results in the previous chapters showed that LIF sampling strongly depends on the accurate choice of neuron parameters. In particular, the choice of time constants is important: The refractory period has to be much larger than the inherent delays on the chip, the synaptic time constant must be of the order of the refractory period and the effective membrane time constant has to be much smaller than the synaptic time constant (see Section 2.1.7). The crucial question in this chapter is therefore: Do the range limitations and the variations of parameters on the HICANN chip suit the needs of LIF sampling? And if not, can suitable workarounds be found?

Section 6.1 thus starts with a characterization of the circuits on the HICANN chip with a particular focus on the parameters which are essential for LIF sampling. First, the values of the floating gate cells are fixed to plausible values needed for LIF sampling and those neurons are selected on the HICANN chip, whose membrane potential exhibits approximately the expected behavior. Thereafter, Spike-Triggered Average (STA) is used to extract the time constants of the selected membrane circuits. It will become clear that the ranges of time constants are unsatisfactory to allow a concrete implementation of LIF sampling. In particular, the saturation of the synaptic conductances and the limited ranges of the synaptic and membrane time constants will complicate the application of LIF sampling on the current version of the HICANN chip.

Section 6.2 therefore aims at providing desired parameter ranges for the LIF neurons based

on the results of software simulations of neurons with the deduced hardware parameters from Section 6.1. It will become apparent that a reasonable LIF sampling performance can only be achieved with larger synaptic time constants together with larger frequencies of the background stimulus or smaller membrane time constants than currently achievable on the used version of the HICANN chip.

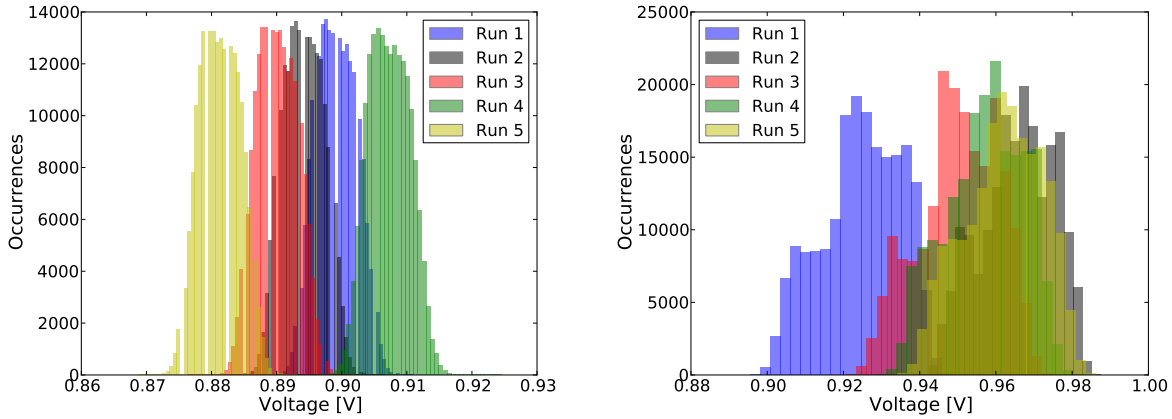
6.1 Characterization of the Neural and Synaptic Circuits of the HICANN chip

This section aims at testing the feasibility of the application of LIF sampling with the neurons integrated on the HICANN chip. Subsection 6.1.1 starts with a description of the utilized set of the individual neural and synaptic parameters. Thereafter, Subsection 6.1.2 continues with a rigorous measurement of the involved time constants, the proportions of which are essential for a satisfactory LIF sampling performance. Subsection 6.1.3 presents the activation curve of a selected neuron and concludes that, with the currently available methods, a generic application of LIF sampling on the current version of the HICANN chip is not feasible.

6.1.1 Parameter Selection

Table A.6 contains the utilized values of the floating gate cells that were used throughout the experiments in this section. The table comprises global and individual neuron parameters as well as parameters of the synaptic circuits. The following paragraphs discuss the choice of the neural and synaptic parameters in the course of this section. Hereby, the hardware parameters are written in `typewriter font`, while their corresponding actual neural and synaptic parameters are written in normal font.

Neural Parameters 24 of the 44 parameters in Table A.6 describe the dynamics of the AdEx membrane circuits (see Section 2.1.6) on the HICANN chip. The neural parameters which were manually set for the experiments in this thesis are the global parameter `V_reset` and the individual neuron parameters `V_t`, `E_l`, `E_synx`, `E_syni`, `I_fire`, `I_gladapt`, `I_gl`, `I_pl`, `V_syntcx` and `V_syntci`.
First of all, the exponential term in Equation 2.40 and the adaptive current $w(t)$ (see Equation 2.41) have to be quiesced such that the membrane dynamics of the neurons approximately reduce to those of standard LIF neurons in Equation 2.34. For this, the DAC-values of the currents `I_fire` and `I_gladapt` are set to 0 [*Millner*, 2012]. However, this does not exclude the existence of remaining leakage currents, such that, for some neurons, adaptive mechanisms are still observable.
The floating gate cells of the leakage potential, `E_l`, and of the reversal potentials, `E_synx` and `E_syni`, are set to 512 DAC, 626 DAC and 398 DAC, respectively, so that the reversal potentials are lying symmetrically with respect to the leakage potential. The leakage and reversal potentials are therefore expected to lie at 0.9 V, 0.7 V and 1.1 V, respectively ($\frac{\text{DAC value}}{1024} \times 1.8\,\text{V}$).

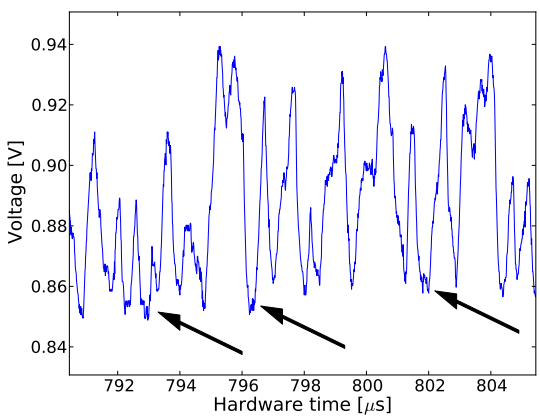


A: Voltage distribution of Neuron #27

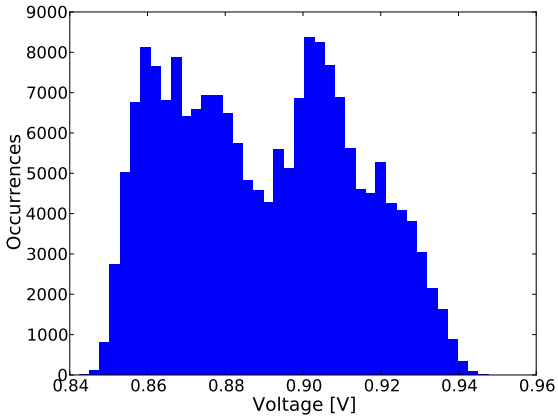
B: Voltage distribution of Neuron #222

Figure 6.1: Distribution of the free membrane voltage of Neuron #27 and #222 on the used HICANN chip. Both panels show the voltage distribution of 5 simulations of 15 s BT with equal parameters. Before each new run, the floating gates are written twice to the values in Table A.6. The expected mean of the voltage distribution is 0.9 V. However, the mean and spread of the distributions varies from neuron to neuron. Panel **A** shows a positive example with a small spread of the membrane potential and small trial-to-trial variations. The spread and the trial-to-trial variations in Panel **B** are unsatisfactory. Empty histogram bins are artifacts which result from the discretization of the analog voltage signal via the ADC.

The values of the floating gate cells `I_gl`, `I_pl`, `V_syntcx` and `V_syntci` determine the membrane time constant, the refractory period and the excitatory and inhibitory synaptic time constants, respectively. The currents `I_gl` and `I_pl` are set to 1000 DAC and 30 DAC, respectively, to ensure small τ_{m} and large τ_{ref}. The voltages `V_syntcx` and `V_syntci` have been fixed to 800 DAC. For values smaller than 800 DAC, the height of the PSP shrinks towards 0 and remains indiscernible from the electrical noise of the membrane potential. For values larger than 800 DAC, the trace of the synaptic conductance shows a saturation plateau (see Figure 6.5).
Furthermore, switching `use_big_capacitors` to `False` allows to choose the smaller one of the two possible membrane capacitances of a neuron block. This ensures that the membrane time constant is as small as possible due to $\tau_{\mathrm{m}} = \frac{C_{\mathrm{m}}}{g_{\mathrm{l}}}$, but has the side effect that the PSPs become larger and saturate faster.
Figure 6.1 shows examples of the distributions of the free membrane potential for the two neurons #27 and #222. Both panels show the voltage distributions of 5 different experimental runs of 15 s BT. For each new run, the floating gate cells are rewritten twice. Both examples are intentionally selected to show the disparity of membrane dynamics of the neurons on the HICANN chip.
The voltage distributions of Neuron #27 in Panel A are close to the expected value 0.9 V



A: Voltage trace of Neuron #27



B: Voltage distribution of Neuron #27

Figure 6.2: Typical voltage trace and voltage distribution of Neuron #27 under a balanced Poisson stimulation with a biological rate of 200 Hz for the excitatory and inhibitory input, respectively. The *black* arrows in Panel **A** indicate saturation effects of the membrane potential, which appear due to the saturation of the synaptic conductances. The voltage distribution results from an experiment of the duration 15 s BT and has a bimodal shape around the mean voltage of about 0.89 V (see Panel **B**). The first peak of the distribution represents the value at which the synaptic conductance, and hence the voltage trace, saturates, while the second peak is at the leakage potential. EPSPs are smaller than IPSPs for the selected parameters and do not saturate.

and Gaussian-shaped due to the noise of the electrical components. Panel B shows the same results for Neuron #222. In comparison to Panel A, the mean value of measured distributions is far from the expected value of 0.9 V and the spread of the distributions is about twice as large as the spread of the distributions of Neuron #27. Moreover, there are large deviations of the mean voltage among the different experimental runs.
In the experiments of the following Subsection 6.1.2, only those neurons on the HICANN chip are considered which show a membrane behavior similar to Neuron #27 regarding the average and spread of the membrane potential and the trial-to-trial variation.

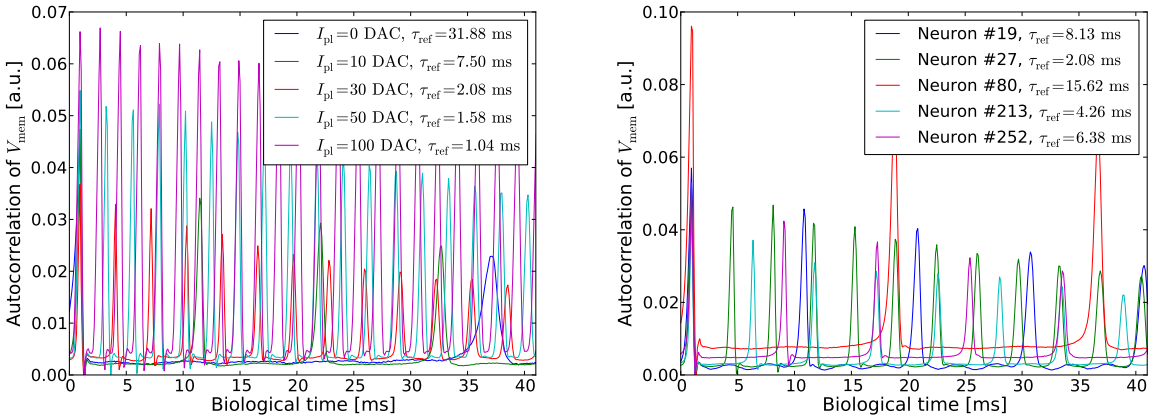
Synaptic Parameters The manually set synaptic parameters are the global parameters `g_max` and `g_max_div` and the individual synaptic weights.
The parameter `g_max` allows the choice of 4 different strengths of the maximal synaptic conductance $g_{\max}$ via an analog floating gate cell [*Schemmel et al.*, 2014]. For all following experiments, `g_max` is set to 0.
The 4-bit parameter `g_max_div` determines the divisor of the maximal conductance. The value of `g_max_div` is set to 5 for the synaptic weights of the background stimulus. This heuristic choice allows, on the one hand, choosing large weights between the neurons,

when setting `g_max_div` to 0, and, on the other hand, prevents the PSPs from becoming very small and thus indistinguishable from the electrical noise on the membrane.
Figure 6.2 shows a typical voltage trace (Panel A) and voltage distribution (Panel B) of a neuron which receives synaptic input from an excitatory and inhibitory Poisson source with a biological rate of 200 Hz, respectively. The spikes are generated on the host computer and sent to the HICANN chip via the FPGA board and the DNC.
Figure 6.2A reveals that the membrane voltage saturates at about 0.86 V which can be traced back to the saturation of the total synaptic conductance, a phenomenon which has also been documented in *Millner* [2012]. The voltage distribution in Figure 6.2B has a bimodal shape around a mean voltage of 0.89 V. The first peak of the distribution at 0.86 V represents the value at which the synaptic conductance, and hence the voltage trace, saturates, while the second peak at 0.89 V is the leakage potential. EPSPs are smaller than IPSPs for the selected parameters and, thus, do not saturate.

6.1.2 Measuring the Time Constants

The appropriate setting of the time constants of a LIF neuron is crucial for the LIF sampling performance (see e.g. Figures 3.4 and 4.4). The refractory period τ_{ref} has to be at least an order of magnitude larger than the inherent synaptic delays, in order to minimize their effect on neuron correlations. The synaptic time constant τ_{syn} has to be in the order of the refractory period τ_{ref}, such that a neuron which elicits a spike has a noticeable impact on the postsynaptic neurons during the next refractory period. In contrast, the effective membrane time constant τ_{eff} of the neuron has to be very small in order to enable fast membrane dynamics. This can be achieved either by setting the membrane time constant τ_{m} to a value close to 0 or by injecting a high-frequency excitatory and inhibitory background stimulus into the neuron.
The following three paragraphs aim at determining the time constants of the neurons on the HICANN chip with a special focus on the consistency with the requirements of LIF sampling. The constants τ_{ref} and τ_{syn} are measured via Spike-Triggered Average (STA) [*de Boer and Kuyper*, 1968], while τ_{m} is measured by injecting a finite current pulse into the membrane and fitting an exponential function to the response of the membrane potential. For each time constant, 200 PSPs (for τ_{syn}), 200 APs (for τ_{ref}), or 200 current pulses (for τ_{m}) are averaged. The experiments are run once with Neuron #27 using different hardware parameters, and once with different neurons which have the same hardware parameters. All measured results of the time constants are given in biological real-time.

Refractory Period For the measurement of τ_{ref}, the hardware parameter `V_t`, which represents the threshold potential, is set to 410 DAC, which is far below the leakage potential. This setting provokes a bursting of the neuron with an ISI of τ_{ref} plus the upswing time until the following spike. Thus, τ_{ref} is the distance between a peak (averaged spike) and the upswing to the following peak. Likewise, the parameter `V_reset` is set to 400 DAC to remain below the threshold potential.

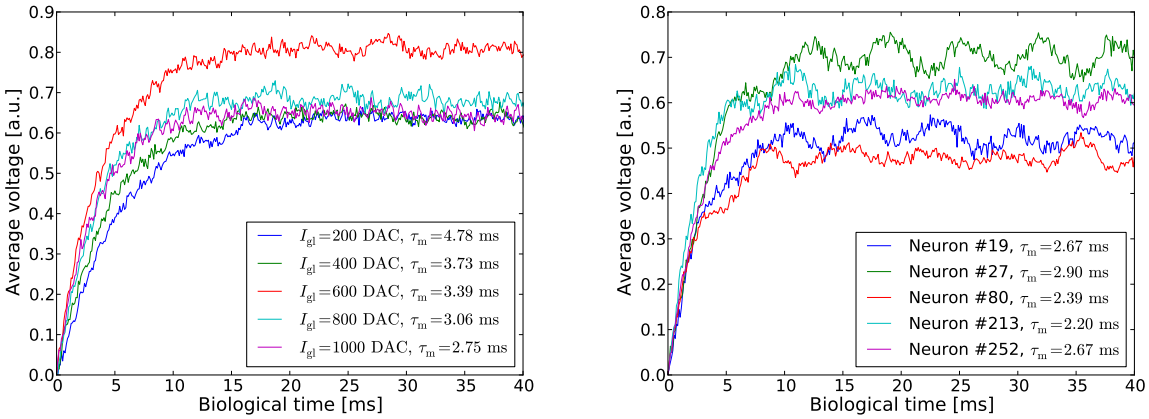


A: Neuron #27 **B:** $I_{pl} = 30\,\text{DAC}$

Figure 6.3: Measurement of the refractory period τ_{ref} of Neuron #27 for different currents `I_pl` in Panel **A** and of different neurons for the current `I_pl` = 30 DAC in Panel **B**. The panels show the autocorrelation of the membrane potential of a neuron whose threshold potential is below the leakage potential, such that the neuron spikes with a ISI close to τ_{ref}. The value of τ_{ref} corresponds to the distance between one peak (spike) and the next upswing. Panel **A** implies that the refractory period can be selected from at least two different orders of magnitude. However, τ_{ref} varies greatly for different neurons when injecting the same currents `I_pl` (see Panel **B**).

Figure 6.3A shows the measured refractory periods τ_{ref} of Neuron #27 for different currents `I_pl`. The figure implies that τ_{ref} can be selected from at least 2 different orders of magnitude: For `I_pl` = 0 DAC, we get $\tau_{ref} = 31.88\,\text{ms}$, while for `I_pl` = 100 DAC, we have $\tau_{ref} = 1.04\,\text{ms}$, which is more than one order of magnitude lower.
Figure 6.3B displays the measured τ_{ref} of different neurons for the current `I_pl` = 30 DAC. Similar to Panel A, the value of τ_{ref} varies by an order of magnitude between 2 ms and 16 ms for different neurons, even if the same current `I_pl` = 30 DAC is selected.
For too low $\tau_{ref} \approx 1.2\,\text{ms}$ the synaptic delays on the HICANN chip, which are fixed to 1.2 ms (see Section 2.2.1), will have a disturbing effect on the resulting LIF sampling quality because they will affect neuron correlations. That is why $\tau_{ref} \gg 1.2\,\text{ms}$ is required and, indeed, achievable for certain neurons on the chip.

Membrane Time Constant The measurement of the membrane time constant τ_m is conducted by averaging the responses of the membrane potential to the same current stimuli and fitting an exponential function to the average curve. The growth constant of the exponential function then corresponds to τ_m.
Figure 6.4A shows the results of the measured τ_m of Neuron #27 for different currents `I_gl`. The membrane time constant can be tuned on a finer scale compared to the

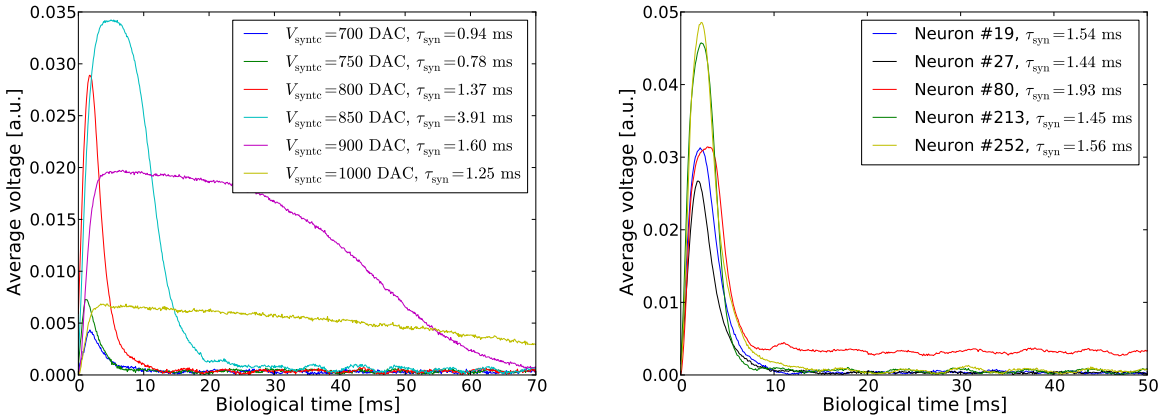


A: Neuron #27 **B:** $I_{gl} = 1000\,\mathrm{DAC}$

Figure 6.4: Measurement of the membrane time constant τ_m of Neuron #27 for different currents `I_gl` in Panel **A** and of different neurons for the current `I_gl` = 1000 DAC in Panel **B**. The membrane time constant was deduced by applying a rectangular current stimulus to the membrane and fitting an exponential function to the upswing of the membrane potential. The neuron-to-neuron variations of τ_m are small compared to these of τ_{ref} in Figure 6.3. The smallest possible value for τ_m is about 2 ms biological time.

refractory period in the last paragraph. Between the currents `I_gl` = 200 DAC and `I_gl` = 1000 DAC, τ_m only changes by a factor of 2.
Also the neuron-to-neuron variations of τ_m in Figure 6.4B are small compared to these of τ_{ref}. For all tested neurons, the smallest value of the membrane time constant amounts to about $\tau_m = 2.0\,\mathrm{ms}$.

Synaptic Time Constant The measurement of the synaptic time constant τ_{syn} is achieved by injecting a regular spike train into the neuron and averaging over the resulting PSPs. By fitting a difference-of-exponentials function to this average PSP, we can deduce the (effective) membrane time constant and the synaptic time constant. Since we know the value of τ_m from the last paragraph, the remaining time constant obtained from the fit belongs to the synaptic conductance.
One important aspect which has to be mentioned at this point is the fact that the PSP shape is not a real difference-of-exponentials function but has an exponential growth at the upswing. However, the difference-of-exponentials function yields a sufficient approximation to the PSP shape.
Figure 6.5A displays the measured synaptic time constant τ_{syn} of Neuron #27 for different voltages `V_syntc`. For each of the displayed curves except for `V_syntc` = 850 DAC, the value of τ_{syn} lies in the range between 0.7 and 1.6 ms. However, for all `V_syntc` > 800 DAC, the PSP saturates at some peak value due to saturating synaptic conductances (see



A: Neuron #27 **B:** $V_{\text{syntc}} = 800\,\text{DAC}$

Figure 6.5: Measurement of the synaptic time constant τ_{syn} of Neuron #27 for different voltages `V_syntc` in Panel **A** and of different neurons for the voltage `V_syntc` = 800 DAC in Panel **B**. For each `V_syntc` except 850 DAC, the value of τ_{syn} lies in the range between 0.7 and 1.6 ms in Panel **A**. However, for each `V_syntc` > 800 DAC, the PSP saturates at some peak value due to saturating synaptic conductances. The synaptic time constant at the voltage `V_syntc` = 800 DAC lies around 1.5 ms for all tested neurons (see Panel **B**).

Subsection 6.1.1). For the remaining range of `V_syntc`, the PSPs can not be distinguished from the electrical noise of the membrane potential.
The synaptic time constant at the voltage `V_syntc` = 800 DAC lies around 1.5 ms for all tested neurons, which is shown in Figure 6.5B. Thus, the synaptic time constant is consistently lower than the membrane time constant of the neurons and of the order of the synaptic delays. The second statement is even worse because τ_{syn} has to be in the range of τ_{ref} (see Chapters 3 and 4) to achieve optimal LIF sampling results, but, however, delays of the order of τ_{ref} will affect neuron correlations.

6.1.3 Measuring the Activation Curve

In the software simulations in Chapters 3, 4 and 5, the activation curve of a LIF neuron was measured by varying the mean membrane potential of the neuron and measuring the fraction p_{ON} of the time that the neuron spends in the refractory state (see Figure 3.1). A shift of the leakage potential towards one of the reversal potentials might introduce non-linearities, additionally to those which are in the system anyway.
Therefore, the method used in this section leaves the leakage potential and the reversal potentials constant such that the reversal potentials remain symmetrical with respect to the leakage potential. In order to trace the activation curve, the threshold voltage `V_t` and the reset voltage `V_reset` = `V_t` − 10 DAC are shifted from the inhibitory towards

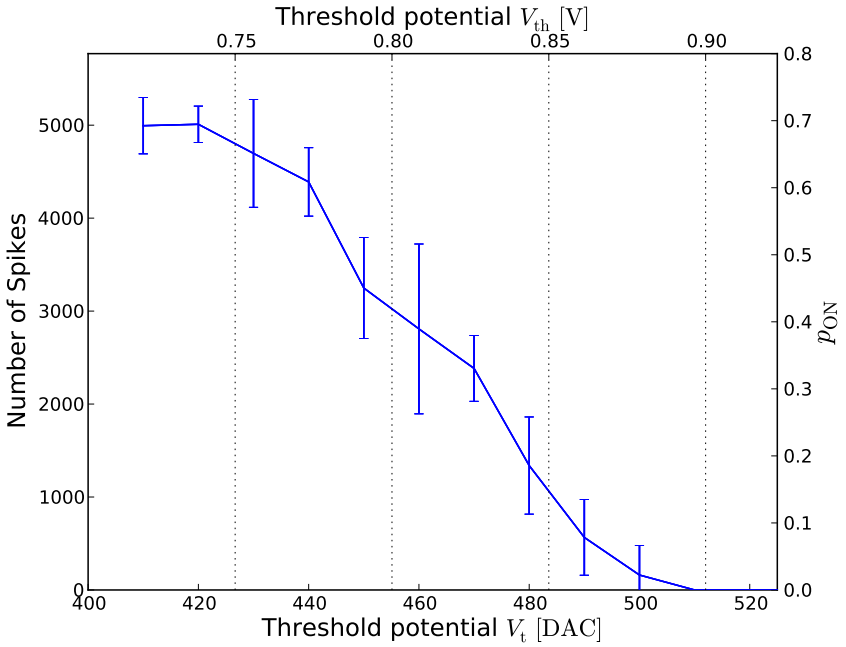


Figure 6.6: Activation curve of Neuron #27. Each measuring point corresponds to the average of 5 experiments of the duration 15 s biological time. The error bars denote the standard deviations. The Poisson background has a biological rate of 200 Hz for the excitatory and inhibitory input, respectively. In opposite to Figure 3.1, the threshold voltage `V_t` and the reset voltage `V_reset` = `V_t`−10 DAC are modified. This ensures that the actual reversal potentials stay symmetrically around the leakage potential. Due to τ_{m} (see Figure 6.4) and the necessary condition `V_t` − `V_reset` ≥ 10 DAC , the neuron maximally reaches $p_{\mathrm{ON}} = 0.7$, which is insufficient for LIF sampling.

the excitatory reversal potential. For each data point, the number N_{S} of elicited spikes during $T = 15\,\mathrm{s}$ biological time is measured. With the extracted refractory period of the neurons from Figure 6.3, $p_{\mathrm{ON}} = \frac{N_{\mathrm{S}}\tau_{\mathrm{ref}}}{T}$ can be calculated.
Figure 6.6 displays the measured activation curve of Neuron #27. Each data point of the curve corresponds to an average over 5 experiments, in which the neuron received an excitatory and inhibitory background input of 200 Hz biological rate[1], respectively. Between consecutive experimental runs, the floating gate cells are reprogrammed twice. The activation curve in Figure 6.6 is sigmoidal and can be well fitted to a logistic function with a proper linear transformation of `V_t`. However, several aspects are unsatisfactory for a reliable application of LIF sampling. First, the curve saturates at a maximum of about 5000 spikes which corresponds to $p_{\mathrm{ON}} \approx 0.7$. This implies that a larger p_{ON} can

[1] A background rate of 200 Hz allows to supply several neurons with the same rate in parallel in prospective experiments with LIF-based BMs, with the given maximal bandwidth and acceleration factor (see Section 2.2.2).

not be reached. This can be traced back to the large τ_{m} and the saturation of synaptic conductances, which results in a large effective membrane time constant $\tau_{\mathrm{eff}} \approx \tau_{\mathrm{m}}$ and thus insufficiently fast membrane dynamics.
A second unsatisfactory issue are the large error bars particularly around the midpoint of the activation curve. These errors can be mainly ascribed to the imprecision of the floating gate cells in the individual experimental runs. However, this inconsistency is expected to be overcome in prospective experiments with a calibrated hardware system. The following subsection will present the results of software simulations which aim to find the parameter values of the time constants and the background input which fit best to the requirements of LIF sampling.

6.2 Towards a Parametrization that Will Enable LIF Sampling on Hardware

The previous section has shown that a naive application of LIF sampling is not directly compatible with the range and the fluctuations of parameters available on the current version of the HICANN chip. This section aims at finding the parameters ranges which fit best to the requirements of LIF sampling, while still taking into account the parameters of the neurons on the HICANN chip, which were deduced in Section 6.1. For this, software simulations are run in which the neurons are initialized with the optimal parameters of the neurons on the HICANN chip. Subsection 6.2.1 deals with the impact of the parameters of the background stimulus on the effective membrane time constant of the LIF neuron. Thereafter, Subsection 6.2.2 investigates the quality of LIF sampling from an example Boltzmann distribution of 5 RVs based on the determined time constants of the neurons on the HICANN chip.

6.2.1 Background Input Parameters

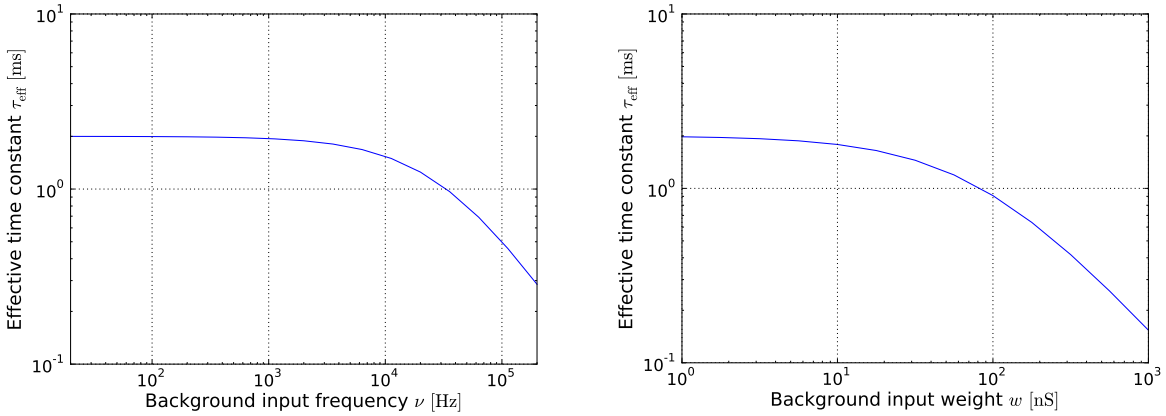
The saturation of the activation function in Figure 6.6 at $p_{\mathrm{ON}} = 0.7$ implies that the effective membrane time constant τ_{eff} is not small enough compared to τ_{syn} and τ_{ref}. This is due to the fact that, on the one hand, the maximum background input frequency and weight to the neuron is too small and, on the other hand, the membrane time constant τ_{m} too large. According to Equation 2.54, both the background input frequency and weight have an impact on the synaptic conductance and, consequently, on the total conductance $g_{\mathrm{tot}} = \frac{C_{\mathrm{m}}}{\tau_{\mathrm{eff}}}$:

$$\overline{g_i(t)} = w_i \nu_i \tau_{\mathrm{syn}} \,. \tag{6.1}$$

With $g_{\mathrm{tot}}(t) = g_l + \sum_i g_i(t)$, the effective membrane time constant τ_{eff} can be directly calculated:

$$\tau_{\mathrm{eff}} = \frac{C_{\mathrm{m}}}{g_{\mathrm{tot}}} = \frac{C_{\mathrm{m}}}{g_{\mathrm{l}} + \tau_{\mathrm{syn}} \left(w_{\mathrm{exc}} \nu_{\mathrm{exc}} + w_{\mathrm{inh}} \nu_{\mathrm{inh}} \right)} \,. \tag{6.2}$$

Now we can fix the parameters $\tau_{\mathrm{syn}} = 1.5\,\mathrm{ms}$ and $\tau_{\mathrm{m}} = 2.0\,\mathrm{ms}$, which were both measured in Section 6.1, and illustrate τ_{eff} as a function of the background input rates $\nu = \nu_{\mathrm{exc}} = \nu_{\mathrm{inh}}$ and weights $w = w_{\mathrm{exc}} = w_{\mathrm{inh}}$.



A: τ_{eff} versus background rate ν

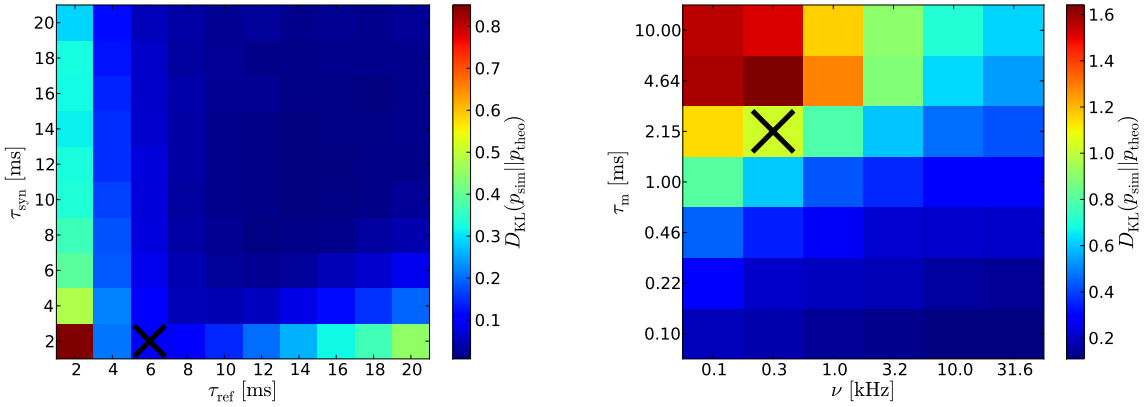
B: τ_{eff} versus background weight w

Figure 6.7: Dependence of the effective membrane time constant τ_{eff} on the rate ν (see Panel **A**) and the weight w (see Panel **B**) of the background Poisson stimulus. The excitatory and inhibitory synaptic weights in Panel **A** are 2 nS. The background rate in Panel **B** amounts to 400 Hz for the excitatory and inhibitory input, respectively.

Figure 6.7A shows the impact of the background input rate ν on τ_{eff} for the synaptic input weights $w = 2.0\,\text{nS}$. For low input frequencies, the effective membrane time constant stays constant at $\tau_{\text{eff}} = \tau_{\text{m}}$. At frequencies higher than 1.0 kHz, τ_{eff} begins to shrink. At about 100.0 kHz and more, τ_{eff} reaches values of 0.2 to 0.4 ms, which are an order of magnitude smaller than τ_{syn}, and, thus, suit the requirements of LIF sampling.
Figure 6.7B displays the dependency of τ_{eff} on the strength of the background input weights w for an input frequency of $\nu = 400.0\,\text{Hz}$ for the excitatory and inhibitory connections, respectively. The plot shows that τ_{eff} decreases with larger synaptic weights. Only for very large synaptic weights close to $w = 1.0\,\mu\text{S}$, τ_{eff} becomes about an order of magnitude smaller than τ_{syn}. However, very large weights are impractical because the neurons would quickly reach the reversal potentials and the PSPs would saturate.

6.2.2 Time Constants

In Section 6.1.2 it was shown that the best achievable values of the time constants on the current version of the HICANN chip are $\tau_{\text{m}} = 2\,\text{ms}$ and $\tau_{\text{syn}} = 1.5\,\text{ms}$. The following discussion aims at finding an optimal working point for LIF sampling for a BM of 5 RVs using the determined hardware parameters. Furthermore, the experiments comprise optimal values for the time constants, which fit best to the requirements of LIF sampling. Figure 6.8 presents the sampling performance based on the KL divergence of the simulated and theoretical joint probability distribution (see Equation 2.15) for a simulated BM of 5 RVs. Each experiment has a duration of 15 s BT, just like in the hardware measurements in Section 6.1. If not given by the color-plots, the parameters of the LIF neuron are



A: $D_{\mathrm{KL}}(p_{\mathrm{sim}}\|p_{\mathrm{theo}})$ versus τ_{ref} and τ_{syn}

B: $D_{\mathrm{KL}}(p_{\mathrm{sim}}\|p_{\mathrm{theo}})$ versus ν and τ_{m}

Figure 6.8: Color-coded KL divergence of the simulated and target joint probability distribution for a simulated BM consisting of 5 RVs. The duration of one simulation is 15 s biological time. Table A.1 shows the utilized neuron and stimulus parameters, but with $\tau_{\mathrm{m}} = 2\,\mathrm{ms}$ in Panel **A** and $\tau_{\mathrm{ref}} = 1.5\,\mathrm{ms}$ and $\tau_{\mathrm{syn}} = 1.5\,\mathrm{ms}$ in Panel **B**. Furthermore, in contrast to Table A.1, the synaptic delays are fixed to 1.2 ms, mimicking the synaptic delays on the HICANN chip. The *black* crosses show the best currently achievable results with the neurons on the HICANN chip. The figures imply that the best sampling results are reached with large τ_{ref} and τ_{syn} compared to τ_{m}, and large input frequencies.

chosen from Table A.1, but with $\tau_{\mathrm{ref}} = 1.5\,\mathrm{ms}$, $\tau_{\mathrm{m}} = 2.0\,\mathrm{ms}$, $\tau_{\mathrm{syn}} = 1.5\,\mathrm{ms}$ and delays of 1.2 ms.

Figure 6.8A shows the sampling quality under the variation of the time constants τ_{syn} and τ_{ref}. The best sampling performance is achieved with the conditions $\tau_{\mathrm{ref}} \gg 1.2\,\mathrm{ms}$, such that the synaptic delay is vanishing compared to the time scales of the neuron, and $\tau_{\mathrm{syn}} \approx \tau_{\mathrm{ref}}$, a result which has already been verified in Chapter 3. The black cross in Figure 6.8A shows the expected best achievable result with the current version of the HICANN chip taking into account a constant synaptic time constant $\tau_{\mathrm{syn}} \approx 2.0\,\mathrm{ms}$.

Figure 6.8B displays the sampling performance for different background input rates ν and membrane time constant τ_{m}. As expected according to Equations 2.48 and 2.55, respectively, the best sampling results are achieved with very large input rates ν and thus a fast membrane, which means $\tau_{\mathrm{eff}} \approx 0$. Due to the bandwidth limitation (see Chapter 2.2) regarding the current acceleration of 10^4 of the HICANN chip compared to biological real-time, however, the best achievable result, when using the parameter set of the neurons on the HICANN chip, is unsatisfactory (see *black cross* in Figure 6.8B).

6.3 Conclusion

This chapter investigated the feasibility of naively implementing LIF sampling on the current version of the HICANN chip. It turned out that the membrane and synaptic circuits on the current version of the HICANN chip do not allow such a direct implementation of LIF sampling for several reasons.
First, and most crucially, the synaptic conductances saturate very quickly. This leads to a rapid saturation of the membrane potential and, consequently, non-additive PSPs.
Second, the synaptic time constant cannot be set larger than the effective membrane time constant, a fact which is crucial for LIF sampling. Due to the short synaptic time constant, the effect of the individual background spikes on the course of the membrane potential of a neuron is very small, demanding for large input frequencies far beyond the maximum bandwidth of about 2775 Hz for regular spike trains at the acceleration factor 10^4 of the HICANN chip compared to biological real-time.
Third, the membrane time constant of the neurons cannot be set smaller than $\tau_\mathrm{m} = 2.0\,\mathrm{ms}$. In the software simulations of the previous chapters, $\tau_\mathrm{m} \approx 0.1\,\mathrm{ms}$ was used, which allowed fast membrane dynamics independent of the background input rates. Software simulations showed that, with the demonstrated constraint of the parameter range of τ_m on the HICANN chip, a very large background input frequency in the range of 10 kHz is required to achieve a fast membrane and, consequently, satisfactory LIF sampling results. This option is, however, not approachable with the current hardware setup due to the bandwidth limitation.
Fourth, imprecisions arise during the measurements which prevent from setting up well-tuned LIF-based BMs. These imprecisions can be be ascribed to the inaccuracies of the floating gate cells. Since many parameters (τ_ref, τ_syn, τ_m, $E^\mathrm{rev}_\mathrm{exc}$, $E^\mathrm{rev}_\mathrm{inh}$, V_rest, V_th) need to be tuned simultaneously, inaccuracies of one parameter affect the precision of other parameters.
All in all, the current version of the HICANN chip appears to prohibit a direct implementation of LIF sampling. In the ongoing experiments, some workarounds can be envisioned, among which mLIF PSPs emerge as a promising candidate, since large τ_ref but only small τ_syn can be achieved.

7 Discussion

The simulation results in this thesis have demonstrated the feasibility of using networks of deterministic Leaky Integrate-and-Fire (LIF) neurons to sample from probability distributions schematized by Bayesian Networks (BNs). For this, two different implementations of BNs over binary RVs, which were proposed by *Pecevski et al.* [2011], have been transferred to the framework of *Petrovici et al.* [2013].
In the first implementation, the BNs were reduced to a Boltzmann Machines (BMs) with the help of auxiliary RVs. Each factor of order larger than 2 of the probability distribution was represented via additional auxiliary RVs, each of which coded for one of the possible states of this higher-order factor. The neurons corresponding to the original RVs satisfied the NCC in the extended probability distribution of the original and auxiliary RVs.
Chapter 4 provided the results of this first implementation of BNs with LIF neurons. The chapter revealed that it is possible to sample from BNs via networks of LIF neurons. However, the sampling quality was unsatisfactory for BNs with an extreme discrepancy of the probability distribution due to the shape of the LIF PSP. In this case, auxiliary neurons with high firing rates consistently silenced auxiliary neurons with very low, non-zero firing rates. This phenomenon arose mainly due to the exponential tail of the PSP shape. A sophisticated modification of the synaptic interaction in order to approach the shape of ideal rectangular PSPs from *Pecevski et al.* [2011] led to a vast improvement of the LIF sampling performance for arbitrary BNs. For this, instead of one neuron per RV, each RV was represented by a feed-forward chain of neurons, which allowed to reshape the PSPs of the postsynaptic sampling neuron. However, similar to the results using the original stochastic neuron model with rectangular PSPs, a slow convergence in the range of hundreds of seconds towards the stationary probability distribution was observed, which was a result of the increase of the probability space dimensionality due to the addition of auxiliary RVs. In contrast, the convergence time with standard LIF PSPs was about an order of magnitude lower, which could be explained by the asymmetry of the PSP shape and, thus, a fast settling towards some stationary distribution.
In the second implementation of BNs with LIF neurons, the principal neurons, which correspond to the original RVs, directly satisfied the NCC. This was achieved by a different network structure which is induced via a Markov blanket expansion of the NCC: Auxiliary neurons were introduced as preprocessing units which code for the states of the Markov blanket of the corresponding principal neuron, while satisfying the NCC. In contrast to the first implementation, the information flow in the neural network was directed, such that concrete distributions of spike timings became crucial.
Chapter 5 described the results of Implementation 2 with LIF neurons. Achieving good sampling quality using both the standard LIF PSPs and the novel mLIF PSPs was particularly challenging compared to the first implementation for three different reasons.

First, the non-uniformity of the PSP shape led to a non-uniformity of the firing probability. Second, the stochasticity induced by the Poisson background input was different from the inherent stochasticity of the abstract neuron model. Third, the autocorrelation of the membrane potential provoked undesired correlations of spike times among the auxiliary neurons. Only for large refractory periods and small synaptic time constants, a modification which came in favor of all three issues, LIF sampling with mLIF PSPs provided adequate results. However, large refractory periods and small synaptic time constants came at the expense of very large neural networks and, therefore, long durations of simulations.

Chapter 6 investigated the feasibility to realize BMs with LIF neurons integrated on the HICANN chip, which is the centerpiece of the neuromorphic hardware system which is being developed in the frame of the *BrainScaleS* project. The neural and synaptic circuits on the current version of HICANN chip, however, appear to not allow a direct implementation of LIF sampling for several reasons. The most crucial issue is the saturation of synaptic conductances due to a production mismatch which leads to a limited dynamic range of the membrane potential. This saturation also does not allow setting the synaptic time constant larger than the membrane time constant, which is required to achieve satisfactory sampling results. Furthermore, the limitation of available independent Poisson sources on the HICANN chip, together with the acceleration factor of 10^4 compared to biological time, only allowed for maximal biological rates in the range of 100 Hz for the Poisson stimulus on the chip. These frequencies, however, were insufficient to facilitate a low effective membrane time constant, and thus, the High-Conductance-State (HCS) required to perform sampling with LIF neurons.

The following chapter will provide an overview of tasks which will extend the existent framework of LIF-based BNs. Furthermore, the potential for facilitating LIF sampling with the current and the new, decelerated version of the HICANN chip, which is currently being developed, is discussed.

8 Outlook

This thesis has successfully demonstrated the applicability of deterministic Leaky Integrate-and-Fire (LIF) neurons for sampling from probability distributions described by Bayesian Networks (BNs). A modification of the standard LIF PSPs, the so-called mLIF PSPs, led to a vast improvement of the sampling quality. The inherent parameter constraints and fluctuations of the neurons and synapses integrated on the BrainScaleS neuromorphic hardware system, however, appeared to not allow a naive application of LIF sampling.
The following discussion contains suggestions for prospective experiments which will enhance the sampling results in both software and hardware implementations.

Improvement of the LIF Sampling Performance in Software Simulations First of all, the mLIF PSP shape needs to be optimized. In particular, the sampling performance has to be measured for different sizes of the feed-forward chains, synaptic time constants and refractory periods.
Second, the LIF sampling performance with mLIF PSPs needs to be tested with respect to the size of the BN. Here, a limiting factor regarding software simulations is expected. The size of the neural networks grows exponentially with the size of the BNs, due to an increasing number of auxiliary RVs. In addition, a larger number of RVs automatically leads to longer convergence times towards the target distributions. Both facts come at the expense of the necessary simulation time.
Third, the question arises, whether the feed-forward chains of neurons might be extended to small populations of neurons in order to increase robustness. This modification will, on the one hand, make the system more complex but, on the other hand, might come in favor of Implementation 2, because the stochastic firing of the neural populations might redistribute the FPTs of the postsynaptic sampling neuron.
Fourth, regarding the simulation results of Implementation 1 when using standard LIF PSPs, a PyNN option facilitating the selection of different synaptic time constants for the connections to the same neuron would allow to choose the shape of the PSP as a function of the weight and, with this, would improve the LIF sampling results.
Furthermore, the work which is planned for the future incorporates the learning of synaptic weights using Contrastive Divergence [*Hinton*, 2002] or rules based on STDP [*Nessler et al.*, 2013], which hold the potential to enhance the LIF sampling results. In addition, as proposed by *Pecevski et al.* [2011], WTA circuits could be used to extend the sampling framework from binary to discrete probability distributions.

Preparation of an Effective Implementation of LIF Sampling on the BrainScaleS Neuromorphic Hardware System Concerning a future implementation of LIF sampling on the HICANN chip, several issues have to be managed. With respect to the current

version of the HICANN chip, one could use the on-chip BEGs and BEGs from neighboring HICANN chips to provide for a high-frequent background input of the hardware neurons, and, consequently, a small effective membrane time constant. Each BEG would be initialized with a different seed, which is randomly updated during the experiment, such that, adding two or more spike trains from two different BEGs might results in a unique Poisson-distributed spike train [*Schemmel*, 2014]. However, the randomness of this spike train needs to be investigated, e.g. by comparing the resulting sampling quality to an ideal Poisson-distributed background spike pattern.
Recurrent networks of spiking neurons, so-called Balanced Random Networks (BRNs), which were investigated in *Jordan* [2013], are a promising candidate to be used for generating uncorrelated background noise for the functional BNs on the neuromorphic hardware substrate, even if the on-chip noise resources are limited. *Jordan* [2013] shows that the output activity of a BRN, whose neurons partly share the same input sources, can be decorrelated by inhibitory feedback within the networks themselves. BRNs integrated on the neuromorphic hardware, on the one hand, will allow a high-frequent background input for the functional networks and, thus, fast membrane dynamics, and on the other hand, will improve the sampling results compared to these with usual Poisson input [*Jordan*, 2013].
The prospective version of the HICANN chip, which is currently being developed, will contain several useful improvements with regard to the feasibility of the implementation of LIF sampling. First, a correct matching of the synaptic conductance will allow for non-saturating membrane voltage traces and additive PSPs, which is a fundamental requirement of LIF sampling. Second, an increase of the reliably achievable refractory times and, by the same amount, of the synaptic constant by at least a magnitude above the inherent synaptic delays on the HICANN chip will come in favor of the LIF sampling results.
The planned reduction of the acceleration factor of the HICANN chip from 10^4 to 10^3 compared to biological time will automatically solve two issues available on the current hardware system regarding the applicability of LIF sampling. On the one hand, the synaptic delays will shrink by an order of magnitude in terms of biological time, thereby becoming less problematic. On the other hand, a reduced speed-up factor will allow for input frequencies, which are by an order of magnitude larger than these on the current hardware system. These high input frequencies are essential to reduce the effective membrane time constant, which will be in favor of the LIF sampling performance.
Last but not least, a complete calibration of the membrane and synaptic circuits will significantly ease the implementation of any network model on the hardware substrate, including, of course, LIF sampling.

A Appendix

A.1 Code References

The code associated with this manuscript is located in the BrainScaleS git repository `git@gitviz.kip.uni-heidelberg.de:model-nmsampling.git`.
The folder `root/code/bayesian` contains the subfolders:

- `neuralsampling`: Implementation 1 and 2 of BMs and BNs using the abstract neuron model
- `impl1LIF`: Implementation 1 with LIF neurons
- `impl2LIF`: Implementation 2 with LIF neurons
- `impl1mLIF`: Implementation 1 with LIF neurons and mLIF PSPs
- `impl2mLIF`: Implementation 2 with LIF neurons and mLIF PSPs
- `Psp`: Design and investigation of PSP shapes

The folder `root/code/gibbs_sampling` provides an efficient implementation of Gibbs sampling from Boltzmann distributions.
Hardware experiments are saved in `root/code/hardware_experiments`:

- `lif_in_hcs`: LIF neuron in a spiking noisy environment provided by spikes created on the host computer and transmitted via the FPGA
- `lif_in_hcs_beg`: LIF neuron in a spiking noisy environment provided by the on-chip BEGs
- `bm_2`: Neural implementation of a BM of 2 RVs
- `bm_3`: Neural implementation of a BM of 3 RVs

A.2 Neuron Parameters

The following tables contain the parameters which have been used for the LIF neurons in the course of this thesis.

Resting membrane potential V_{rest}	$\overline{V_k^b}$ with bias b
Capacity of the membrane C_{m}	0.2 nF
Membrane time constant τ_{m}	0.1 ms
Duration of refractory period τ_{ref}	30.0 ms
Decay time of the excitatory synaptic conductance $\tau_{\text{syn,exc}}$	30.0 ms
Decay time of the inhibitory synaptic conductance $\tau_{\text{syn,inh}}$	30.0 ms
Reversal potential for excitatory input $E_{\text{exc}}^{\text{rev}}$	0.0 mV
Reversal potential for inhibitory input $E_{\text{inh}}^{\text{rev}}$	-100.0 mV
Spike threshold V_{th}	-50.0 mV
Reset potential after a spike V_{reset}	-53.0 mV
Offset current I_{offset}	0.0 nA
Utilization of synaptic efficacy U_0	1.0
Recovery time constant τ_{rec}	$0.99 \cdot 30.0$ ms
Facilitation time constant τ_{facil}	0.0 ms
Excitatory/inhibitory Poisson input rate	400.0 Hz/400.0 Hz
Excitatory/inhibitory background weight	0.002 μS/0.002 μS
Synaptic delays	0.1 ms

Table A.1: Standard parameters of the LIF neurons with conductance-based synapses utilized in Chapter 3.

Resting membrane potential V_{rest}	$\overline{V_k^b}$ with bias b
Capacity of the membrane C_{m}	0.2 nF
Membrane time constant τ_{m}	0.1 ms
Duration of refractory period τ_{ref}	20.0 ms
Decay time of the excitatory synaptic conductance $\tau_{\text{syn,exc}}$	10.0 ms
Decay time of the inhibitory synaptic conductance $\tau_{\text{syn,inh}}$	10.0 ms
Reversal potential for excitatory input $E_{\text{exc}}^{\text{rev}}$	0.0 mV
Reversal potential for inhibitory input $E_{\text{inh}}^{\text{rev}}$	-100.0 mV
Spike threshold V_{th}	-50.0 mV
Reset potential after a spike V_{reset}	-53.0 mV
Offset current I_{offset}	0.0 nA
Utilization of synaptic efficacy U_0	1.0
Recovery time constant τ_{rec}	$0.99 \cdot 10.0$ ms
Facilitation time constant τ_{facil}	0.0 ms
Excitatory/inhibitory Poisson input rate	400.0 Hz/400.0 Hz
Excitatory/inhibitory background weight	0.002 μS/0.002 μS
Synaptic delays	0.1 ms

Table A.2: Standard parameters of the LIF neurons with conductance-based synapses utilized in Chapter 4 and 5.

Resting membrane potential V_{rest}	-52.3 mV
Capacity of the membrane C_{m}	0.2 nF
Membrane time constant τ_{m}	0.1 ms
Duration of refractory period τ_{ref}	19.8 ms
Decay time of the excitatory synaptic conductance $\tau_{\text{syn,exc}}$	2.0 ms
Decay time of the inhibitory synaptic conductance $\tau_{\text{syn,inh}}$	2.0 ms
Reversal potential for excitatory input $E_{\text{exc}}^{\text{rev}}$	0.0 mV
Reversal potential for inhibitory input $E_{\text{inh}}^{\text{rev}}$	-100.0 mV
Spike threshold V_{th}	-50.0 mV
Reset potential after a spike V_{reset}	-52.3 mV
Offset current I_{offset}	0.0 nA
Synaptic delays	0.1 ms

Table A.3: Parameters of the principal LIF neurons utilized in Chapter 5.

Parameters of the sampling neurons	
Capacity of the membrane C_m	0.2 nF
Membrane time constant τ_m	0.1 ms
Duration of refractory period τ_ref	29.5 ms
Decay time of the excitatory synaptic conductance $\tau_\mathrm{syn,exc}$	30.0 ms
Decay time of the inhibitory synaptic conductance $\tau_\mathrm{syn,inh}$	30.0 ms
Reversal potential for excitatory input $E_\mathrm{exc}^\mathrm{rev}$	0.0 mV
Reversal potential for inhibitory input $E_\mathrm{inh}^\mathrm{rev}$	-100.0 mV
Spike threshold V_th	-50.0 mV
Reset potential after a spike $V_\mathrm{reset}^\mathrm{mLIF\text{-}50}$	-50.01 mV
Reset potential after a spike $V_\mathrm{reset}^\mathrm{mLIF\text{-}53}$	-53.00 mV
Parameters of the forwarding neurons	
Capacity of the membrane C_m	0.2 nF
Membrane time constant τ_m	0.1 ms
Duration of refractory period τ_ref	29.3 ms
Decay time of the excitatory synaptic conductance $\tau_\mathrm{syn,exc}$	2.0 ms
Decay time of the inhibitory synaptic conductance $\tau_\mathrm{syn,inh}$	2.0 ms
Reversal potential for excitatory input $E_\mathrm{exc}^\mathrm{rev}$	0.0 mV
Reversal potential for inhibitory input $E_\mathrm{inh}^\mathrm{rev}$	-100.0 mV
Spike threshold V_th	-50.0 mV
Reset potential after a spike V_reset	-52.3 mV
Resting membrane potential V_rest	-52.3 mV
Parameters of the neural chain	
Number of chain neurons	6
Delay: sampling → sampling neuron	0.1 ms
Delay: sampling → forwarding neuron	5.8 ms
Delay: forwarding → sampling neuron	0.1 ms
Delay: forwarding → forwarding neuron	5.8 ms
Delay: forwarding → last forwarding neuron	5.9 ms
Weight: sampling → sampling neuron	w
Weight: sampling → forwarding neuron	0.16 μS
Weight: forwarding → sampling neuron	$0.180 \cdot w$
Weight: last forwarding → sampling neuron	$-0.815 \cdot w$
Weight: forwarding → forwarding neuron	0.16 μS

Table A.4: Parameters of the chain of 6 neurons which is used to create mLIF PSPs in Chapters 4 and 5

Parameters of the sampling neurons	
Capacity of the membrane C_m	0.2 nF
Membrane time constant τ_m	0.1 ms
Duration of refractory period τ_ref	99.0 ms
Decay time of the excitatory synaptic conductance $\tau_\mathrm{syn,exc}$	10.0 ms
Decay time of the inhibitory synaptic conductance $\tau_\mathrm{syn,inh}$	10.0 ms
Reversal potential for excitatory input $E_\mathrm{exc}^\mathrm{rev}$	0.0 mV
Reversal potential for inhibitory input $E_\mathrm{inh}^\mathrm{rev}$	-100.0 mV
Spike threshold V_th	-50.0 mV
Reset potential after a spike V_reset	-50.01 mV
Parameters of the forwarding neurons	
Capacity of the membrane C_m	0.2 nF
Membrane time constant τ_m	0.1 ms
Duration of refractory period τ_ref	98.8 ms
Decay time of the excitatory synaptic conductance $\tau_\mathrm{syn,exc}$	2.0 ms
Decay time of the inhibitory synaptic conductance $\tau_\mathrm{syn,inh}$	2.0 ms
Reversal potential for excitatory input $E_\mathrm{exc}^\mathrm{rev}$	0.0 mV
Reversal potential for inhibitory input $E_\mathrm{inh}^\mathrm{rev}$	-100.0 mV
Spike threshold V_th	-50.0 mV
Reset potential after a spike V_reset	-52.3 mV
Resting membrane potential V_rest	-52.3 mV
Parameters of the neural chain	
Number of chain neurons	166
Delay: sampling $\rightarrow$ sampling neuron	0.1 ms
Delay: sampling $\rightarrow$ forwarding neuron	0.5 ms
Delay: forwarding $\rightarrow$ sampling neuron	0.1 ms
Delay: forwarding $\rightarrow$ forwarding neuron	0.5 ms
Weight: sampling $\rightarrow$ sampling neuron	w
Weight: sampling $\rightarrow$ forwarding neuron	0.16 μS
Weight: forwarding $\rightarrow$ sampling neuron	$0.0583 \cdot w$
Weight: last forwarding $\rightarrow$ sampling neuron	$-0.9430 \cdot w$
Weight: forwarding $\rightarrow$ forwarding neuron	0.16 μS

Table A.5: Parameters of the chain of 166 neurons which is used to create mLIF PSPs in Chapter 5

Floating gate parameters of the membrane circuits			
Parameter	**Value [DAC]**	**Parameter**	**Value [DAC]**
I_breset	1023	V_reset (global)	590
I_bstim	1023	E_l	512
V_bout	1023	E_syni	398
V_bexp	1023	E_synx	626
V_br	0	I_bexp	0
V_bstdf	0	I_convi	1023
V_ccas	800	I_convx	1023
V_clrc	0	I_fire	0
V_clra	0	I_gl	1023
V_dep	0	I_gladapt	0
V_fac	0	I_intbbi	1023
V_dllres	200	I_intbbx	1023
V_dtc	0	I_pl	30
V_gmax0	1000	I_radapt	1023
V_gmax1	1000	I_rexp	1023
V_gmax2	1000	I_spikeamp	1023
V_gmax3	1000	V_exp	1023
V_m	0	V_syni	1023
int_op_bias	1023	V_syntci	800
V_stdf	0	V_syntcx	800
V_thigh	0	V_synx	1023
V_tlow	0	V_t	600
Parameters of the synapse circuits			
Parameter	**Value**	**Parameter**	**Value**
g_max	0	g_max_div	5
Exc. synaptic weight	5	Inh. synaptic weight	5
Additional settings			
Parameter	**Value**	**Parameter**	**Value**
fg_speed_up_scaling	NORMAL	use_big_capacitors	False

Table A.6: Floating gate parameters of the membrane and synapse circuits on the HICANN chip which have been used in Chapter 6. The membrane parameters on the left hand side are global, the membrane parameters on the right hand side are individual neuron parameters except for the global parameter V_{reset}.

A.3 Acronyms

ADC Analog-to-Digital-Converter

AdEx Adaptive Exponential Integrate-and-Fire

ANC Analog Network Core

AP Action Potential

API Application Programming Interface

ARQ Automatic Repeat reQuest

ASIC Application Specific Integrated Circuit

BEG Background Event Generator

BM Boltzmann Machine

BN Bayesian Network

BRN Balanced Random Network

BT Biological Time

CMOS Complementary Metal Oxide Semiconductor

DAC Digital Analog Converter

DenMem Dendrite membrane

DNC Digital Network Chip

EPSP Excitatory Postsynaptic Potential

FACETS Fast Analog Computing with Emergent Transient States

FPGA Field Programmable Gate Array

FPT First-Passage Time

HAL Hardware Abstraction Layer

HCS High-Conductance-State

HICANN High Input Count Analog Neural Network

HMF Hybrid Multiscale Facility

HT Hardware Time

IPSP Inhibitory Postsynaptic Potential

ISI	Inter-Spike Interval
KL	Kullback-Leibler
L1	Layer 1
L2	Layer 2
LFSR	Linear Feedback Shift Register
LIF	Leaky Integrate-and-Fire
LVDS	Low Voltage Differential Signaling
MCMC	Markov chain Monte Carlo
mLIF	"Multiple" Leaky Integrate-and-Fire
NCC	Neural computability condition
ODE	Ordinary Differential Equation
OU	Ornstein-Uhlenbeck
PCB	Printed Circuit Board
PDF	Probability Density Function
PSP	Postsynaptic Potential
RV	Random Variable
SEB	System Emulator Board
SRAM	Static Random-Access Memory
STA	Spike-Triggered Average
STDP	Spike-Timing Dependent Plasticity
STP	Short-Term Plasticity
TM	Tsodyks-Markram
TNN	Two-Node-Network
UMC	United Microelectronics Corporation
VLSI	Very-Large-Scale Integration
VPE	Visual Perception Experiment
WTA	Winner-Take-All

Bibliography

4th BrainScaleS Demo3 workshop, Fourth BrainScaleS Demo3 workshop, Fuerberg, Austria, May 29-31, 2013.

Abrahams, D., and R. Grosse-Kunstleve, Building hybrid systems with Boost.Python, 2003.

Ackley, D. H., G. E. Hinton, and T. J. Sejnowski, A learning algorithm for Boltzmann machines, *Cognitive Science*, *9*, 147–169, 1985.

Alais, D., and R. Blake, *Binocular Rivalry*, A Bradford book, MIT Press, 2005.

Arbib, M. A. (Ed.), *The Handbook of Brain Theory and Neural Networks*, 2nd ed., MIT Press, Cambridge, MA, USA, 2002.

Berkes, P., G. Orbán, M. Lengyel, and J. Fiser, Spontaneous cortical activity reveals hallmarks of an optimal internal model of the environment., *Science*, *331*, 83–7, 2011.

Bishop, C. M., *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

BrainScaleS, Research, `http://brainscales.kip.uni-heidelberg.de`.

Brette, R., and W. Gerstner, Adaptive exponential integrate-and-fire model as an effective description of neuronal activity, *J. Neurophysiol.*, *94*, 3637 – 3642, 2005.

Brette, R., et al., Simulation of networks of spiking neurons: A review of tools and strategies, *Journal of Computational Neuroscience*, *23*, 349–398, 2007.

Brüderle, D., E. Müller, A. Davison, E. Muller, J. Schemmel, and K. Meier, Establishing a novel modeling tool: A python-based interface for a neuromorphic hardware system, *Front. Neuroinform.*, *3*, 2009.

Brüderle, D., et al., A comprehensive workflow for general-purpose neural modeling with highly configurable neuromorphic hardware systems, *Biological Cybernetics*, *104*, 263–296, 2011.

Brunel, N., and S. Sergi, Firing frequency of integrate-and-fire neurons with finite synaptic time constants, *Journal of Theoretical Biology*, *195*, 87–95, 1998.

Buesing, L., J. Bill, B. Nessler, and W. Maass, Neural dynamics as sampling: A model for stochastic computation in recurrent networks of spiking neurons, *PLoS Computational Biology*, *7*, e1002,211, 2011.

Bytschok, I., From shared input to correlated neuron dynamics: Development of a predictive framework, Diploma thesis, University of Heidelberg, HD-KIP 11-87, 2011.

Davison, A. P., D. Brüderle, J. Eppler, J. Kremkow, E. Muller, D. Pecevski, L. Perrinet, and P. Yger, PyNN: a common interface for neuronal network simulators, *Front. Neuroinform.*, *2*, 2008.

Dayan, P., and L. F. Abbott, *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*, The MIT press, Cambride, Massachusetts, 2001.

de Boer, R., and P. Kuyper, Triggered correlation., *IEEE Trans Biomed Eng*, *15*, 169–179, 1968.

Destexhe, A., M. Rudolph, and D. Pare, The high-conductance state of neocortical neurons in vivo, *Nature Reviews Neuroscience*, *4*, 739–751, 2003.

Diesmann, M., and M.-O. Gewaltig, NEST: An environment for neural systems simulations, in *Forschung und wisschenschaftliches Rechnen, Beiträge zum Heinz-Billing-Preis 2001*, edited by T. Plesser and V. Macho, vol. 58 of *GWDG-Bericht*, pp. 43–70, Ges. für Wiss. Datenverarbeitung, Göttingen, 2002.

Doya, K., S. Ishii, A. Pouget, and R. Rao, *Bayesian Brain: Probabilistic Approaches to Neural Coding*, Computational Neuroscience Series, MIT Press, 2011.

Electronic Vision(s), Website, `http://www.kip.uni-heidelberg.de/vision`.

Friedmann, S., A new approach to learning in neuromorphic hardware, Ph.D. thesis, Heidelberg, Univ., Diss., 2013, 2013.

Geman, S., and D. Geman, Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, *IEEE Trans. Pattern Anal. Machine Intell*, *6*, 721–741, 1984.

Gerstner, W., and W. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*, Cambridge University Press, 2002.

Gleeson, P., et al., Neuroml: A language for describing data driven models of neurons and networks with a high degree of biological detail., *PLoS Computational Biology*, *6*, 2010.

Goodman, D., and R. Brette, Brian: a simulator for spiking neural networks in Python, *Front. Neuroinform.*, *2*, 2008.

Gotoh, J.-y., H. Jin, and U. Sumita, Numerical evaluation of dynamic behavior of ornstein–uhlenbeck processes modified by various boundaries and its application to pricing barrier options, *Methodology and Computing in Applied Probability*, *13*, 193–219, 2011.

Griffiths, T. L., and J. B. Tenenbaum, Optimal predictions in everyday cognition, *Psychological Science 17*, 2006.

Griffiths, T. L., C. Kemp, and J. B. Tenenbaum, Bayesian models of cognition, *Cambridge Handbook of Computational Cognitive Modeling*, 2008.

Grimmett, G., and D. Stirzaker, *Probability and random processes*, vol. 80, Oxford university press, 2001.

Hartmann, S., S. Schiefer, S. Scholze, J. Partzsch, C. Mayr, S. Henker, and R. Schuffny, Highly integrated packet-based aer communication infrastructure with 3gevent/s throughput, in *Electronics, Circuits, and Systems (ICECS), 2010 17th IEEE International Conference on*, pp. 950 –953, 2010.

Hines, M., and N. Carnevale, Neuron simulation environment, in *The Handbook of Brain Theory and Neural Networks*, pp. 769–773, M.A. Arbib, 2003.

Hinton, G. E., Training products of experts by minimizing contrastive divergence, *Neural Comput.*, *14*, 1771–1800, 2002.

Hinton, G. E., Boltzmann machine, *Scholarpedia*, *2*, 1668, 2007.

Hopfield, J. J., Neural networks and physical systems with emergent collective computational abilities, *Proceedings of the National Academy of Sciences*, *79*, 2554–2558, 1982.

Ide, J. S., and F. G. Cozman, Random generation of bayesian networks, in *In Brazilian Symp. On Artificial Intelligence*, pp. 366–375, Springer-Verlag, 2002.

Jordan, J., Deterministic recurrent networks as a source of uncorrelated noise for functional neural system, Master's thesis, Forschungszentrum Jülich, 2013.

Knill, D. C., and D. Kersten, Apparent surface curvature affects lightness perception, 1991.

Knill, D. C., and A. Pouget, The bayesian brain: the role of uncertainty in neural coding and computation., *Trends Neurosci*, *27*, 712–719, 2004.

Koller, D., and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*, MIT Press, 2009.

Kononov, A., Testing of an analog neuromorphic network chip, Diploma thesis (English), University of Heidelberg, HD-KIP-11-83, 2011.

Lauritzen, S., and D. Spiegelhalter, Local computations with probabilities on graphical structures and their application to expert systems, *Journal of the Royal Statistical Society – Series B*, *50*, 157–224, 1988.

Levin, D. A., Y. Peres, and E. L. Wilmer, *Markov chains and mixing times*, American Mathematical Society, 2006.

Markram, H., Y. Wang, and M. Tsodyks, Differential signaling via the same axon of neocortical pyramidal neurons., *Proceedings of the National Academy of Sciences of the United States of America*, *95*, 5323–5328, 1998.

Mead, C. A., *Analog VLSI and Neural Systems*, Addison Wesley, Reading, MA, 1989.

Mead, C. A., and M. A. Mahowald, A silicon model of early visual processing, *Neural Networks*, *1*, 91–97, 1988.

Millner, S., Development of a multi-compartment neuron model emulation, 2012.

Nessler, B., M. Pfeiffer, L. Buesing, and W. Maass, Bayesian computation emerges in generic cortical microcircuits through spike-timing-dependent plasticity, *PLOS Computational Biology*, *9*, e1003,037, 2013.

Oaksford, M., and N. Chater, *Bayesian Rationality: The Probabilistic Approach to Human Reasoning*, Oxford Cognitive Science Series, OUP Oxford, 2007.

Pearl, J., Bayesian networks: A model of self-activated memory for evidential reasoning, in *Proceedings of the 7th Conference of the Cognitive Science Society, University of California, Irvine*, pp. 329–334, 1985.

Pearl, J., *Probabilistic reasoning in intelligent systems: networks of plausible inference*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

Pecevski, D., L. Buesing, and W. Maass, Probabilistic inference in general graphical models through sampling in stochastic networks of spiking neurons, *PLoS Computational Biology*, *7*, 2011.

Pecevski, D. A., T. Natschläger, and K. N. Schuch, Pcsim: A parallel simulation environment for neural circuits fully integrated with Python, *Front. Neuroinform.*, *3*, 2009.

Peterson, L. L., and B. S. Davie, *Computer Networks: A Systems Approach, 3rd Edition*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

Petkov, V., Toward belief propagation on neuromorphic hardware, Diploma thesis (English), Ruprecht-Karls-Universität Heidelberg, HD-KIP 12-23, 2012.

Petrovici, M. A., J. Bill, I. Bytschok, J. Schemmel, and K. Meier, Stochastic inference with deterministic spiking neurons., *CoRR*, *abs/1311.3211*, 2013.

PyNN, A Python package for simulator-independent specification of neuronal network models – website, `http://www.neuralensemble.org/PyNN`, 2014.

Python, The Python Programming Language – website, `http://www.python.org`, 2014.

Ricciardi, L., *Diffusion processes and related topics in biology*, Lecture notes in biomathematics, Springer-Verlag, 1977.

Schemmel, J., personal communication, 2014.

Schemmel, J., A. Grübl, K. Meier, and E. Muller, Implementing synaptic plasticity in a VLSI spiking neural network model, in *Proceedings of the 2006 International Joint Conference on Neural Networks (IJCNN)*, IEEE Press, 2006.

Schemmel, J., J. Fieres, and K. Meier, Wafer-scale integration of analog neural networks, in *Proceedings of the 2008 International Joint Conference on Neural Networks (IJCNN)*, 2008.

Schemmel, J., D. Brüderle, A. Grübl, M. Hock, K. Meier, and S. Millner, A wafer-scale neuromorphic hardware system for large-scale neural modeling, in *Proceedings of the 2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1947–1950, 2010.

Schemmel, J., S. Friedmann, A. Grübl, S. Jeltsch, A. Kononov, S. Millner, E. Müller, and B. Vogginger, Specification of the HICANN microchip, FACETS project internal documentation, 2014.

Schemmel, J., et al., Live demonstration: A scaled-down version of the BrainScaleS wafer-scale neuromorphic system, 2012.

Schwartz, M.-O., *PhD thesis*, University of Heidelberg, in preparation, 2012.

Sejnowski, T. J., Higher-order boltzmann machines, in *Neural Networks for Computing*, pp. 398–403, American Institute of Physics, 1986.

Song, S., K. Miller, and L. Abbott, Competitive hebbian learning through spike-timing-dependent synaptic plasticity, *Nat. Neurosci.*, *3*, 919–926, 2000.

Srowig, A., J.-P. Loock, K. Meier, J. Schemmel, H. Eisenreich, G. Ellguth, and R. Schüffny, Analog floating gate memory in a 0.18 μm single-poly CMOS process, *FACETS internal documentation*, 2007.

Steimer, A., W. Maass, and R. J. Douglas, Belief propagation in networks of spiking neurons, *Neural Computation*, *21*, 2502–2523, 2009.

Tsodyks, M., and H. Markram, The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability, *Proceedings of the national academy of science USA*, *94*, 719–723, 1997.

Uhlenbeck, G. E., and L. S. Ornstein, On the theory of the brownian motion, *Phys. Rev.*, *36*, 823–841, 1930.

Acknowledgments (Danksagungen)

Allen voran Mihai und Ilja für die osum Betreuung.

Herrn Meier und Johannes für die dritte Aufnahme bei den Visions.

Allen Visionären für die freundliche Atmosphäre.

Meiner one and only Motivationskünstlerin Nina.

Mama.

Ferdi und Waldi.

Irina, Wowa, Leon, Oma und Alex.

Adriana, Degga und Lara sowie die Eingesaarländerten Andi, André, Fabe, Hans, Julian, Paul und Raphi für die (rückblickend doch so) schöne Studienzeit.

Den Gräther Kickers, the gräthest team of all time: 1 - Bieri, 1 - Nilles, 5 - Pueppi, 7 - Gill, 8 - Dany Gee, 9 - Marv, 10 - Pippo, 11 - Jens, 13 - Kosta, 14 - Lukki, 17 - Konzka, ? - Motschi.

Paulé.

Janine.

Der Fußballgruppe des IWR.

Statement of Originality (Erklärung):

I certify that this thesis, and the research to which it refers, are the product of my own work. Any ideas or quotations from the work of other people, published or otherwise, are fully acknowledged in accordance with the standard referencing practices of the discipline.

Ich versichere, daß ich diese Arbeit selbständig verfaßt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, March 28, 2014

.....
(signature)